



**Calhoun: The NPS Institutional Archive**  
**DSpace Repository**

---

Theses and Dissertations

1. Thesis and Dissertation Collection, all items

---

1976

A study of algorithms to compute the discrete Fourier transform and sensitivity considerations when implemented with sampled analog devices.

Pollack, Michael Anthony

Monterey, California. Naval Postgraduate School

---

<http://hdl.handle.net/10945/17956>

---

*Downloaded from NPS Archive: Calhoun*



Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

**Dudley Knox Library / Naval Postgraduate School**  
**411 Dyer Road / 1 University Circle**  
**Monterey, California USA 93943**

<http://www.nps.edu/library>

A STUDY OF ALGORITHMS TO COMPUTE THE  
DISCRETE FOURIER TRANSFORM  
AND SENSITIVITY CONSIDERATIONS WHEN  
IMPLEMENTED WITH  
SAMPLED ANALOG DEVICES

Michael Anthony Pollack

TEMP. 117  
INSTRUMENT REPORT

DUDLEY KNOX LIBRARY.  
NAVAL POSTGRADUATE SCHOOL  
MONTEREY, CALIF. 93940

INTERNALLY DISTRIBUTED

REPORT

NAVAL POSTGRADUATE SCHOOL

Monterey, California



# THESIS

A STUDY OF ALGORITHMS TO COMPUTE THE  
DISCRETE FOURIER TRANSFORM  
AND SENSITIVITY CONSIDERATIONS WHEN  
IMPLEMENTED WITH  
SAMPLED ANALOG DEVICES

by

Michael Anthony Pollack

June 1976

Thesis Advisor:

T.F. Tao

Approved for public release; distribution unlimited.

T174981



DUDLEY  
NAVAL F  
MONTE

## REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS  
BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Study of Algorithms to Compute the Discrete Fourier Transform and Sensitivity Considerations when Implemented with Sampled Analog Devices		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; June 1976
7. AUTHOR(s) Michael Anthony Pollack		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE June 1976
		13. NUMBER OF PAGES 183
		15. SECURITY CLASS. (of this report)  Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Sampled Analog Devices Fourier Transform Fast Fourier Transform Chirp-z-Transform Prime Transform		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The Fast Fourier Transform (FFT) algorithm has been the most attractive procedure to carry out digital Discrete Fourier Transform (DFT) analysis. However, interest has been generated in two other algorithms, the chirp-z-transform (CZT) and the prime transform (PT), because they are well suited for hardware implementation using the large scale integration (LSI) technologies for sampled analog devices		

INTERNALLY DISTRIBUTED  
REPORT



( ABSTRACT continued)

in spectral analysis applications. Digital computer programs for carrying out the CZT and PT were developed. Their results were compared with results computed by the FFT. The sensitivity of their analysis to parametric variations of their hardware implementations was studied by examining their impulse, rectangular, and composite sinusoidal input responses. Variations in the multiplier and in the transversal tap weights were considered.



A Study of Algorithms to Compute the Discrete Fourier  
Transform and Sensitivity Considerations when Implemented  
with Sampled Analog Devices

by

Michael Anthony Pollack  
Lieutenant, United States Navy Reserve  
E.S.E.F., West Virginia University, 1970

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the  
NAVAL POSTGRADUATE SCHOOL  
June 1976



## ABSTRACT

The Fast Fourier Transform (FFT) algorithm has been the most attractive procedure to carry out digital Discrete Fourier Transform (DFT) analysis. However, interest has been generated in two other algorithms, the chirp-z-transform (CZT) and the prime transform (PT), because they are well suited for hardware implementation using the large scale integration (LSI) technologies for sampled analog devices in spectral analysis applications. Digital computer programs for carrying out the CZT and PT were developed. Their results were compared with results computed by the FFT. The sensitivity of their analysis to parametric variations of their hardware implementations was studied by examining their impulse, rectangular, and composite sinusoidal input responses. Variations in the multiplier and in the transversal tap weights were considered.





## TABLE OF CONTENTS

I. INTRODUCTION.....	12
A. BACKGROUND.....	12
B. BASIC CONCEPTS OF FOURIER TRANSFORM ANALYSIS.....	14
C. IMPLEMENTATION PROBLEMS.....	22
1. Aliasing.....	22
2. Leakage.....	24
D. SPECTRUM ANALYZER PERFORMANCE FIGURES OF MERIT.....	33
1. Sampling Frequency and Anti-Aliasing Filter.....	33
2. Bandwidth (Frequency Range).....	33
3. Resolution or 3-dB Bandwidth.....	34
4. Noise Bandwidth.....	34
5. Highest Sidelobe Level and Asymptotic Roll-off.....	34
6. Single-Sided Spectra.....	35
E. SPECTRAL ANALYSIS ALGORITHMS.....	37
1. The Fast Fourier Transform.....	37
a. General Development of the FFT Algorithm.....	37
b. Decimation-In-Time (DIT) Algorithm...	40
2. The Chirp-Z-Transform (CZT).....	42
a. The Derivation of the CZT Algorithm..	42
3. The Prime Transform.....	51
a. The Prime Transform Derivation.....	51
F. HARDWARE AND SOFTWARE IMPLEMENTATION.....	55
1. Analog Spectral Analysis.....	55
a. Scanning Analyzers.....	55
b. Multifilter Spectrum Analyzers.....	56
2. Digital Spectral Analysis.....	56



a. Technology.....	58
b. Algorithm.....	59
c. Hardware Architecture.....	59
3. Sampled Analog Spectral Analysis.....	62
II. FAST FOURIER TRANSFORM ALGORITHMS.....	66
A. MATHEMATICAL ILLUSTRATION OF THE DECIMATION-IN-TIME (DIT) ALGORITHM.....	66
B. DECIMATION-IN-FREQUENCY (DIF) ALGORITHM.....	74
C. EXAMPLES USING THE FFT ALGORITHM.....	76
III. CHIRP-Z-TRANSFORM COMPUTER SIMULATION.....	83
A. CZT PROGRAM DEVELOPMENT.....	83
B. EXAMPLES USING THE CZT ALGORITHM.....	88
IV. PRIME TRANSFORM COMPUTER SIMULATION.....	96
A. PRIME TRANSFORM COMPUTER PROGRAM DEVELOPMENT.....	96
B. EXAMPLES USING THE PRIME TRANSFORM ALGORITHM.....	100
V. CZT SENSITIVITY ANALYSIS.....	107
A. DIGITAL FFT SYSTEM.....	107
B. SAMPLED ANALOG CZT SYSTEM.....	109
C. SENSITIVITY TESTS.....	110
VI. CONCLUSIONS AND RECOMMENDATIONS.....	153
Appendix A: ALGORITHM TEST DATA.....	154
Appendix B: BRIEF REVIEW OF INTEGERS AND INDICES.....	157
Appendix C: PROGRAM LISTING OF THE FFT ALGORITHM.....	161
Appendix D: PROGRAM LISTING OF THE CZT ALGORITHM.....	163
Appendix E: PROGRAM LISTING OF THE PRIME TRANSFORM ALGORITHM.....	172
LIST OF REFERENCES.....	180
INITIAL DISTRIBUTION LIST.....	183



## LIST OF TABLES

I.	Properties of the Fourier Transform .....	17
II.	Data Windows .....	32
III.	Comparison of Major Logic Families .....	57
IV.	Multiplier Comparison Table .....	58
V.	Required Arithmetic Operations .....	60
VI.	Comparison of Processor Organizations .....	61
VII.	Comparison of Signal Processors .....	65



## LIST OF FIGURES

1. Fourier Transform Pairs.....	18
2. Fourier Transform of a Waveform Sampled at the Nyquist Sampling Rate.....	22
3. Fourier Transform of a Waveform Sampled at Less Than the Nyquist Sampling Rate.....	23
4. DFT of a Band-limited Periodic Waveform: Truncation Equal to an Integer Multiple.....	25
5. DFT of a Periodic Waveform: Truncation Interval Not Equal to an Integer Multiple.....	27
6. Expanded Illustration of the Convolution of Figure [5e].....	28
7. The Weighting Function Process.....	28
8. Hanning Function Fourier Transform Pair.....	30
9. Multiplications Required to Compute the DFT.....	39
10. Contour of the CZT in the Z-Plane.....	44
11. Contour of the CZT in the S-Plane.....	45
12. Processing Operations of the CZT Algorithm.....	49
13. FFT Signal Flowgraph, $N=4$ .....	71
14. Decimation-In-Time Butterfly.....	72
15. FFT Canonical Flow Graphs.....	73
16. Decimation-In-Frequency Butterfly.....	74
17. Magnitude Plot Via FFT Algorithm, Case 1.....	77





18. Phase Plct Via FFT Algorithm, Case 1.....	78
19. Magnitude Plot Via FFT Algorithm, Case 2.....	79
20. Phase Plct Via FFT Algorithm, Case 2.....	80
21. Magnitude Plot Via CZT Algorithm, Case 3.....	81
22. Phase Plct Via FFT Algorithm, Case 3.....	82
23. DFT Via CZT Algorithm With Parallel Implementation of Complex Arithmetic.....	87
24. Magnitude Plot Via CZT Algorithm, Case 1.....	90
25. Phase Plct Via CZT Algorithm, Case 1.....	91
26. Magnitude Plot Via CZT Algorithm, Case 2.....	92
27. Phase Plot Via CZT Algorithm, Case 2.....	93
28. Magnitude Plot Via CZT Algorithm, Case 3.....	94
29. Phase Plot Via CZT Algorithm, Case 3.....	95
30. Functional Diagram of the Prime Transform Algorithm.....	99
31. Magnitude Plot Via Prime Transform Algorithm, Case 1.....	101
32. Phase Plct Via Prime Transform Algorithm, Case 1....	102
33. Magnitude Plot Via Prime Transform Algorithm, Case 2.....	103
34. Phase Plct Via Prime Transform Algorithm, Case 2....	104
35. Magnitude Plot Via Prime Transform Algorithm, Case 3.....	105
36. Phase Plot Via Prime Transform Algorithm, Case 3....	106
37. Impulse Input (normalized A/N).....	112



38.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=0\%$ .....	113
39.	Time Impulse Response of the Sine Filter; $N=512$ , $p=0\%$ .....	114
40.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=4\%$ .....	115
41.	Time Impulse Response of the Sine Filter; $N=512$ , $p=4\%$ .....	116
42.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=10\%$ .....	117
43.	Time Impulse Response of the Sine Filter; $N=512$ , $p=10\%$ .....	118
44.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=20\%$ .....	119
45.	Time Impulse Response of the Sine Filter; $N=512$ , $p=20\%$ .....	120
46.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=30\%$ .....	121
47.	Time Impulse Response of the Sine Filter; $N=512$ , $p=30\%$ .....	122
48.	Time Impulse Response of the Cosine Filter; $N=512$ , $p=40\%$ .....	123
49.	Time Impulse Response of the Sine Filter; $N=512$ , $p=40\%$ .....	124
50.	Magnitude Plot of Input Impulse, $p=0\%$ .....	125
51.	Magnitude Plot of Input Impulse, $p=4\%$ .....	126
52.	Magnitude Plot of Input Impulse, $p=10\%$ .....	127
53.	Magnitude Plot of Input Impulse, $p=20\%$ .....	128



54.	Magnitude Plot of Input Impulse, $p=30\%$ .....	129
55.	Magnitude Plot of Input Impulse, $p=40\%$ .....	130
56.	Rectangular Pulse Input.....	133
57.	Magnitude Plot of Rectangular Pulse Input, $p=0\%$ ....	134
58.	Magnitude Plot of Rectangular Pulse Input, $p=4\%$ ....	135
59.	Magnitude Plot of Rectangular Pulse Input, $p=10\%$ ....	136
60.	Magnitude Plot of Rectangular Pulse Input, $p=20\%$ ....	137
61.	Magnitude Plot of Rectangular Pulse Input, $p=30\%$ ....	138
62.	Magnitude Plot of Rectangular Pulse Input, $p=40\%$ ....	139
63.	Sinusoidal Input.....	140
64.	Magnitude Plot of the Sinusoidal Input, $p=0\%$ .....	141
65.	Phase Plot of the Sinusoidal Input, $p=0\%$ .....	142
66.	Magnitude Plot of the Sinusoidal Input, $p=4\%$ .....	143
67.	Phase Plot of the Sinusoidal Input, $p=4\%$ .....	144
68.	Magnitude Plot of the Sinusoidal Input, $p=10\%$ .....	145
69.	Phase Plot of the Sinusoidal Input, $p=10\%$ .....	146
70.	Magnitude Plot of the Sinusoidal Input, $p=20\%$ .....	147
71.	Phase Plot of the Sinusoidal Input, $p=20\%$ .....	148
72.	Magnitude Plot of the Sinusoidal Input, $p=30\%$ .....	149
73.	Phase Plot of the Sinusoidal Input, $p=30\%$ .....	150
74.	Magnitude Plot of the Sinusoidal Input, $p=40\%$ .....	151
75.	Phase Plot of the Sinusoidal Input, $p=40\%$ .....	152



## I. INTRODUCTION

### A. BACKGROUND

The Fourier transform has been with us many years and the uniqueness of the transform allows Fourier analysis techniques developed in one field to be applied to many diverse areas. Typical application areas of the Fourier transform include:

Linear Systems - The Fourier transform of the output of a linear system is given by the product of the system transfer function and the Fourier transform of the input signal.

Antennas - The field pattern of an antenna is given by the Fourier transform of the antenna current illumination.

Optics - Optical systems have the property that a Fourier transform relation exists between the light amplitude distribution at the front and back focal planes of a converging lens.

Random Process - The power density spectrum of a random process is given by the Fourier transform of the auto-correlation function of the process.

Probability - The characteristic function of a random variable is defined as the Fourier transform of the probability density function of the random variable.

Quantum Physics - The "uncertainty principle" in quantum theory is fundamentally associated with the Fourier transform since particle momentum and position are essentially related through the Fourier transform.

Boundary-Value Problems - The solution of partial differential equations can be obtained by means of the Fourier transform.





Although Fourier analysis allows one to more easily examine a function from another point of view, i.e., the transform domain, the methods available before 1965 to compute the Fourier coefficients were very time consuming and costly. Then in 1965 Cooley and Tukey published their mathematical algorithm which computed the Fourier coefficients with much less computation effort than had been required in the past. This method has become known as the "fast Fourier transform" or FFT and has produced major changes in computational techniques used in digital spectral analysis, filter simulation and related fields.

Although the FFT is the most well known algorithm to compute the Fourier coefficients, it is not the only method for computing the "discrete Fourier transform" (DFT). The chirp-z-transform (CZT) is an algorithm for evaluating the z-transform of a finite duration sequence along certain general contours in the z-plane. Evaluating the DFT of the sequence is a special case of the CZT, i.e., the z-transform of the finite duration sequence is evaluated on the unit circle in the z-plane. Although the CZT is not quite as efficient as the FFT, it eliminates many of the restrictions of the latter. These restrictions are listed at the end of the CZT algorithm derivation. The "prime transform" is an algorithm that follows directly from the CZT derivation. It is based upon "N", the number of samples of data, being an odd prime.

The CZT and prime transform algorithms are receiving considerable attention for applications in spectral analysis. This has been brought about because the bulk of the computations of these algorithms is performed by a transversal filter, which is easily implemented by charge transfer devices (CTD) or surface acoustic wave (SAW) devices. Hardware and software implementations of the FFT,



CZT, and prime transform algorithms are discussed later in this chapter.

## B. BASIC CONCEPTS OF FOURIER TRANSFORM ANALYSIS

The essence of the Fourier transform of a waveform is to decompose or separate the waveform into a sum of sinusoids of different frequencies. The pictorial representation of the Fourier transform is a diagram which displays the amplitude and frequency of each of the determined sinusoids.

Mathematically, the AMPLITUDE SPECTRAL DENSITY or more generally the Fourier transform of  $x(t)$  is given by the relationship

$$X(f) = \int_{-\infty}^{\infty} x(t) e^{-j2\pi ft} dt \quad (1-1)$$

where  $x(t)$  is the waveform to be decomposed into a sum of sinusoids,  $X(f)$  is the Fourier transform of  $x(t)$ , and  $j = \sqrt{-1}$ . The INVERSE FOURIER TRANSFORM is given by the relationship

$$x(t) = \int_{-\infty}^{\infty} X(f) e^{+j2\pi ft} df \quad (1-2)$$

The Fourier transform and its inverse may conveniently be written as

$$v(t) \longleftrightarrow V(f)$$

FOURIER TRANSFORM PAIR



For discrete signal processing the equations take on the following form. For a periodic function  $x(t)$  the Fourier transform is defined by

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j(2\pi/N)nk}, \quad k=0,1,\dots,N-1 \quad (1-3)$$

The "true" frequency component is given by

$$X(k/NT) = \sum_{n=0}^{N-1} x(nT) e^{-j(2\pi/N)nk} \quad (1-4)$$

Eqs. [1-3] and [1-4] are known as the DISCRETE FOURIER TRANSFORM (DFT). The inverse DFT is given by the following relation

$$x(n) = (1/N) \sum_{k=0}^{N-1} X(k) e^{+j(2\pi/N)nk}, \quad n=0,1,\dots,N-1 \quad (1-5)$$

and for "true" time

$$x(rT) = (1/N) \sum_{k=0}^{N-1} X(k/NT) e^{+j(2\pi/N)nk} \quad (1-6)$$



where

$N \rightarrow$  total number of samples of time  
and samples of frequency

$T \rightarrow$  sample period

$n \rightarrow$  sample period index

$k \rightarrow$  harmonic index

$F_c = (1/NT) \rightarrow$  fundamental frequency

For future reference, the basic properties of the continuous Fourier transform and discrete Fourier transform are summarized in Table-I. Fig.[1] gives a pictorial representation of some of the most commonly used Fourier transform pairs.





TABLE-I Properties of the Fourier Transform

Fourier Transform	Property	Discrete Fourier Transform
$x(t)+y(t) \Leftrightarrow X(f)+Y(f)$	Linearity	$x(n)+y(n) \Leftrightarrow X(k)+Y(k)$
$H(t) \Leftrightarrow h(-f)$	Symmetry	$\frac{1}{N}H(n) \Leftrightarrow h(-k)$
$h(t-t_0) \Leftrightarrow H(f)e^{-j2\pi ft_0}$	Time shifting	$h(n-i) \Leftrightarrow H(k)e^{-j2\pi ki/N}$
$h(t)e^{j2\pi ft_0} \Leftrightarrow H(f-f_0)$	Frequency shifting	$h(n)e^{j2\pi ni/N} \Leftrightarrow H(k-i)$
$\left[ \int_{-\infty}^{\infty} H^*(f)e^{-j2\pi ft} df \right]^*$	Alternate inversion formula	$\left[ \frac{1}{N} \sum_{k=0}^{N-1} H(k)e^{-j2\pi kn/N} \right]^*$
$h_e(t) \Leftrightarrow R_e(f)$	Even functions	$h_e(n) \Leftrightarrow R_e(k)$
$h_o(t) \Leftrightarrow jI_o(f)$	Odd functions	$h_o(n) \Leftrightarrow jI_o(k)$
$h(t)=h_e(t)+h_o(t)$	Decomposition	$h(n)=h_e(n)+h_o(n)$
$= \left[ \frac{h(t)}{2} + h\left(-\frac{t}{2}\right) \right] + \left[ \frac{h(t)}{2} - h\left(-\frac{t}{2}\right) \right]$		$= \left[ \frac{h(n)}{2} + \frac{h(N-n)}{2} \right] + \left[ \frac{h(n)}{2} - \frac{h(N-n)}{2} \right]$
$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t-\tau)d\tau = x(t)*h(t)$	Convolution	$y(n) = \sum_{i=0}^{N-1} x(i)h(n-i) = x(n)*h(n)$
$y(t)*h(t) \Leftrightarrow Y(f)H(f)$	Time convolution theorem	$y(n)*h(n) \Leftrightarrow Y(k)H(k)$
$y(t) = \int_{-\infty}^{\infty} x(\tau)h(t+\tau)d\tau$	Correlation	$y(n) = \sum_{i=0}^{N-1} x(i)h(n+i)$
$y(t)h(t) \Leftrightarrow Y(f)*H(f)$	Frequency convolution theorem	$y(n)h(n) \Leftrightarrow \frac{1}{N} Y(k)*H(k)$
$\int_{-\infty}^{\infty} h^2(t)dt = \int_{-\infty}^{\infty}  H(f) ^2 df$	Parseval's theorem	$\sum_{n=0}^{N-1} h^2(n) = \frac{1}{N} \sum_{k=0}^{N-1}  H(k) ^2$



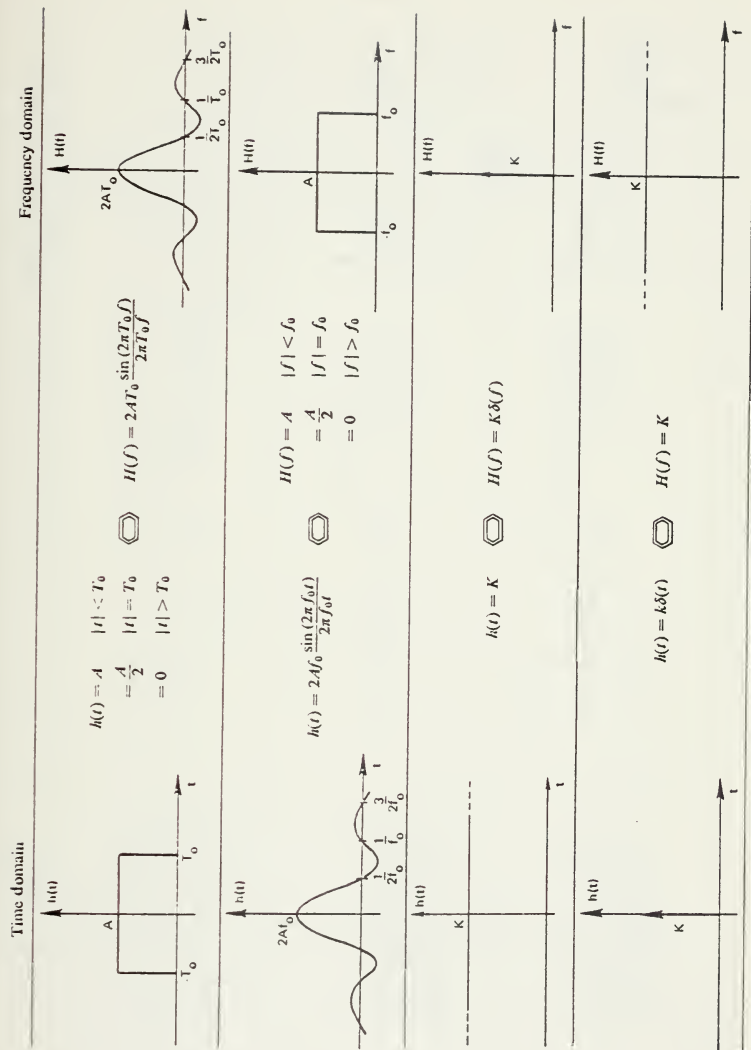


Figure 1 - FOURIER TRANSFORM PAIRS



Time domain

Frequency domain

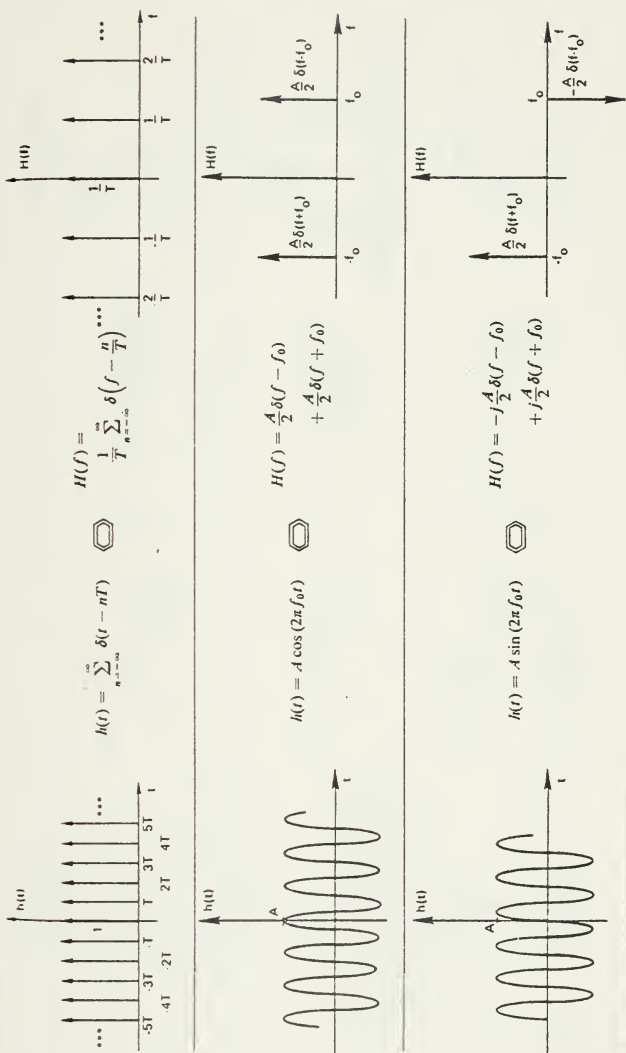


Figure 1 - CONTINUED



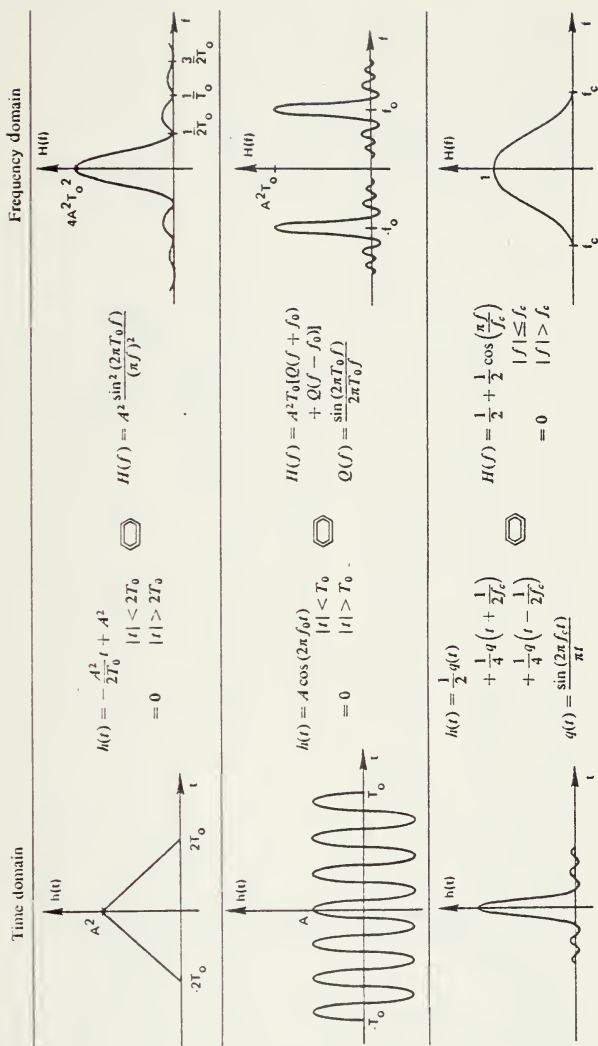


Figure 1-. CONTINUED





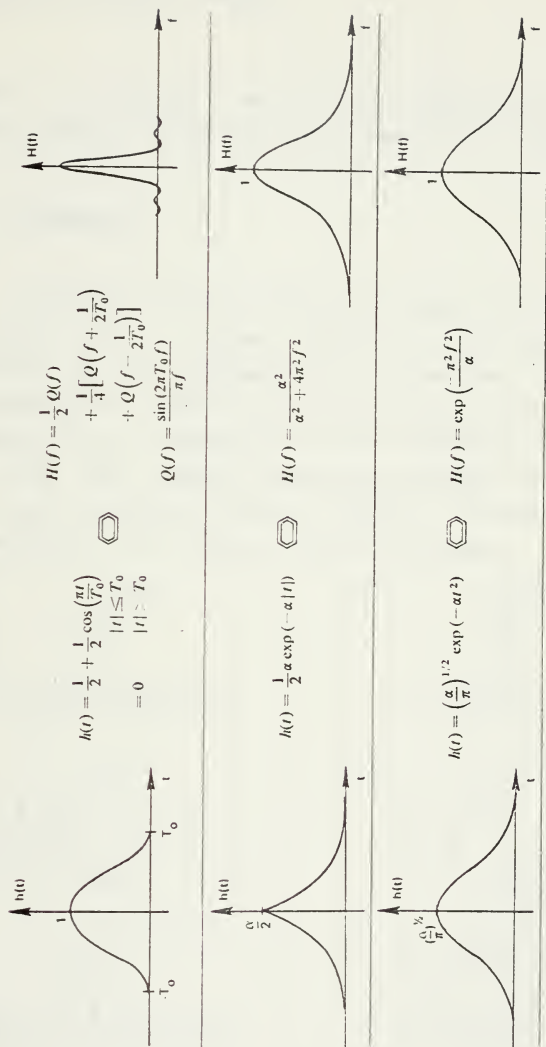


Figure 1 - CONTINUED



## C. IMPLEMENTATION PROBLEMS

The two problems most often encountered in using the discrete Fourier transform are aliasing and leakage.

### 1. Aliasing

Aliasing refers to the fact that the high-frequency components of a time function can impersonate low frequencies if the sampling rate is too low. This problem is corrected by sampling the signal at a rate  $\geq$  the Nyquist Frequency, i.e., demanding that the sampling rate be high enough for the highest frequency present to be sampled at least twice during the period. Fig.[2] gives a pictorial representation of a signal without aliasing.

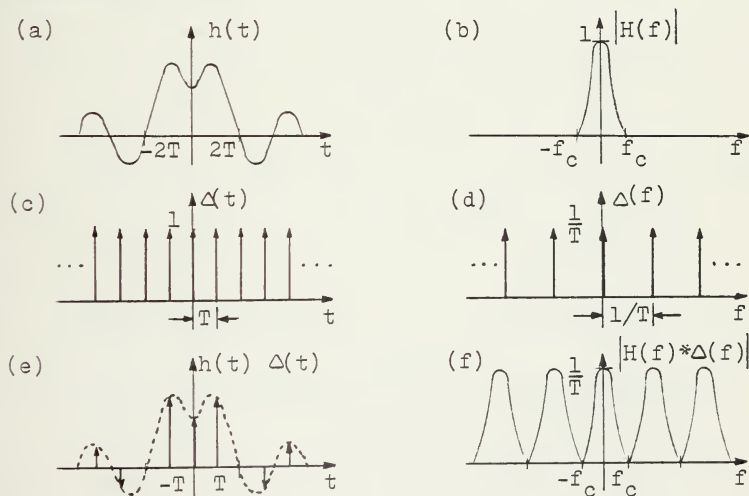


Figure 2 - Fourier Transform of a Waveform Sampled at the Nyquist Sampling Rate.



When the sampling period  $T$  of Fig.[2-c] is increased as shown in Fig.[3-c], the equidistant impulses of  $(f)$  become more closely spaced, Fig.[3-d]. Because of the decreased spacing of the frequency impulses, their convolution with the frequency function,  $H(f)$ , Fig.[3-b] results in the overlapping waveform illustrated results in the overlapping waveform illustrated in Fig.[3-f]. This overlap is the result of aliasing by the high frequency components.

The aliasing phenomenon will occur anytime a sampled analog signal contains frequency components greater than half the sampling frequency,  $(f_s)/2$ . The aliasing frequency relationship is such that all frequencies higher than the analysis bandwidth [ $f_{\max} = (f_s)/2$ ] "fold" down into that band in an accordion-like manner with a fold at every multiple of  $f_{\max}$ .

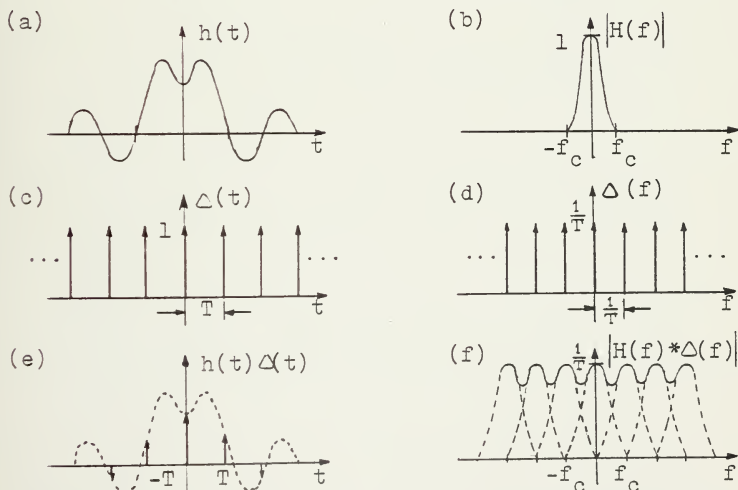


Figure 3 - Fourier Transform of a Waveform Sampled at Less Than the Nyquist Sampling Rate.



## 2. Isakace

If a periodic , band limited function is sampled and truncated to consist of other than an integer multiple of the period, the resulting discrete and continuous Fourier transforms will differ considerably. This problem is inherent in the Fourier analysis of any finite record of data. The record has been formed by looking at the actual signal for T seconds and by neglecting everything that happened before or after this period. As shown in Fig.[4-c], this is equivalent to multiplying the signal by a rectangular window.

If the continuous Fourier transform of the pure cosine wave of Fig.[4-a] had been found, its contribution would have been limited to a pair of impulse functions on the frequency axis, Fig.[4-a].





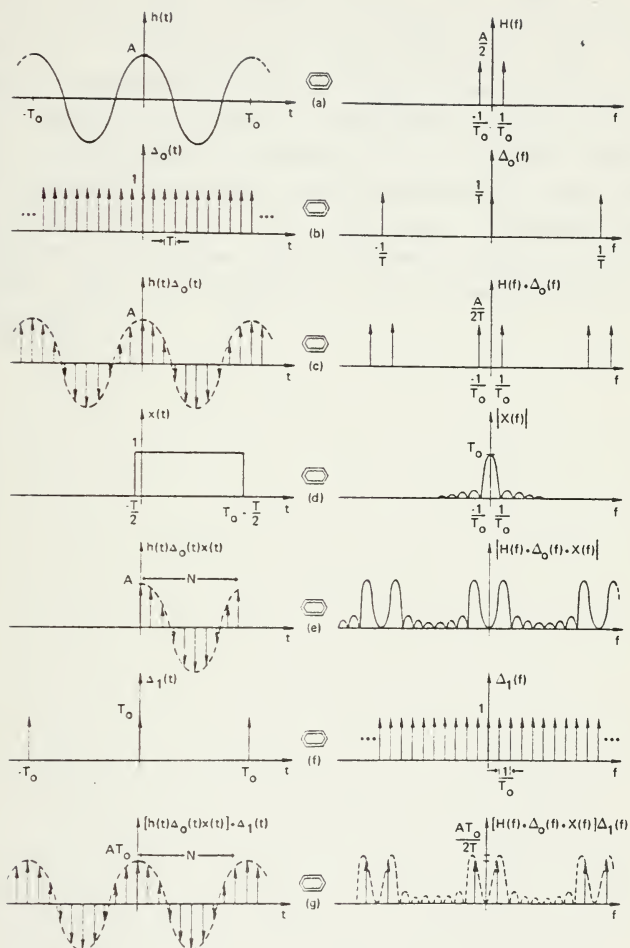


Figure 4 - DFT of a Band-limited Periodic Waveform: Truncation Equal to an Integer Multiple.



The effect of truncation at other than a multiple of the period of the sampled signal is to create a periodic function with sharp discontinuities as shown in Fig.[5]. The multiplication by the data window in the time domain is equivalent to performing convolution in the frequency domain. Consequently, the frequency function is no longer a single impulse but rather a continuous function of frequency with a local maximum centered at the original impulse and a series of spurious peaks termed sidelobes. These sidelobes are responsible for the additional frequency components "leakage", which occur after frequency domain sampling. An expanded view of sampling in the frequency domain is shown in Fig.[6].



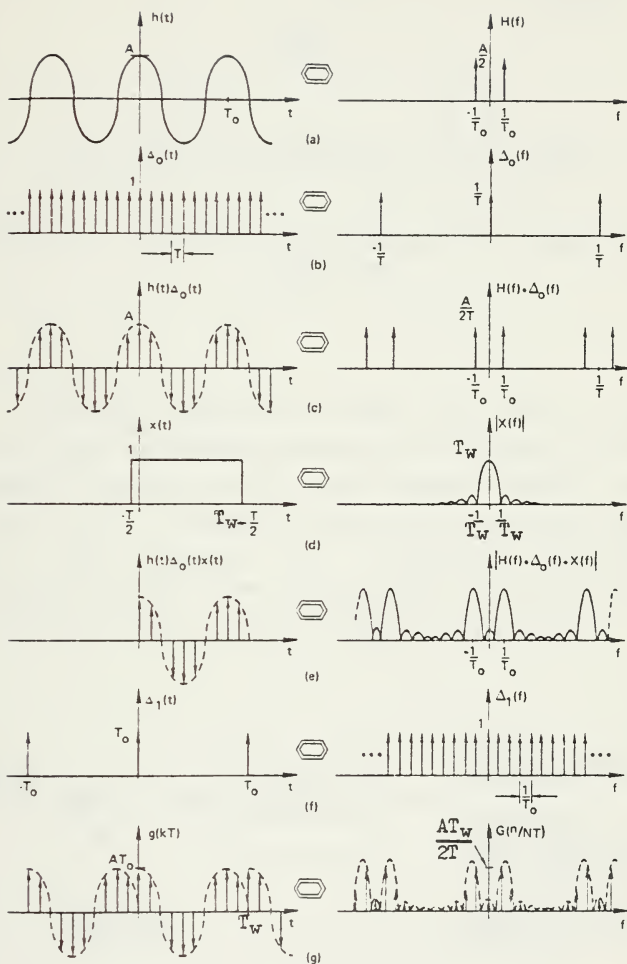


Figure 5 - DFT of a Periodic Waveform: Truncation Interval  
Not Equal to an Integer Multiple



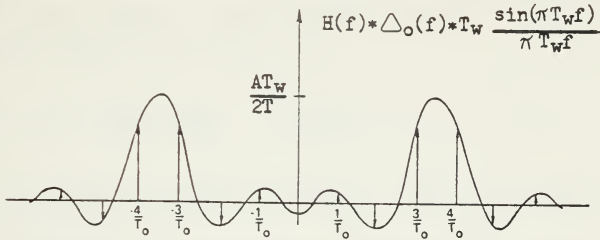


Figure 6 - Expanded Illustration of the Convolution of Figure [5e].

The usual approach to reduce the amount of "leakage" through these sidelobes consists of applying a data window or weighting function to the time series, which has lower sidelobes in the frequency domain than the rectangular data window. The process of weighting consists of amplitude-modulating the signal prior to Fourier analysis. The weighting process is illustrated in Fig.[7].

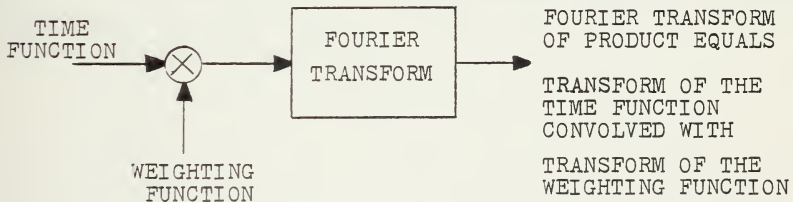


Figure 7 - The Weighting Function Process.





There are many such windows but one of the simplest and most often used is the Hanning function. This function is a cosine-squared curve given by

$$w(t) = \frac{1}{2} - \frac{1}{2} \cos 2\pi \frac{t}{T_c} \quad 0 \leq t \leq T_c \quad (1-7)$$

where  $T_c$  is the truncation interval. The magnitude of the Fourier transform of the Hanning function is given by

$$|W(f)| = \frac{1}{2} Q(f) + \frac{1}{4} \left[ Q\left(f + \frac{1}{T_c}\right) + Q\left(f - \frac{1}{T_c}\right) \right] \quad (1-8)$$

where  $Q(f) = [\sin(\pi T_c f)] / \pi f$

The Hanning function Fourier transform pair is shown in Fig.[8].

Windowing of the input frame by the Hanning function reduces "leakage" by imposing a quasi-periodicity on the input signal and eliminating the discontinuities between periods. The Hanning function widens the main lobe of the  $(\sin x)/x$  shape and reduces the spectral amplitude by a little more than 4dB, but it causes the "interfering" sidelobes to fall off at 18dB per octave instead of 6dB as in the rectangular window.



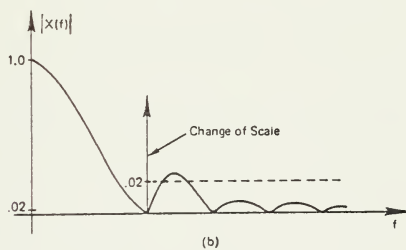
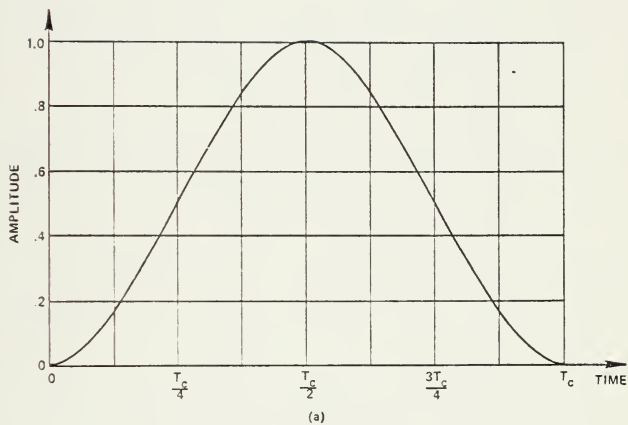


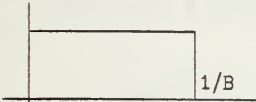

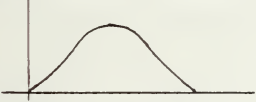
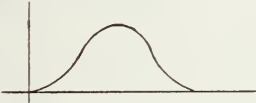
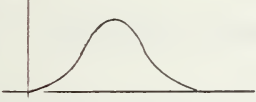
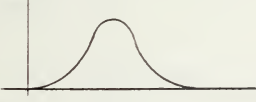
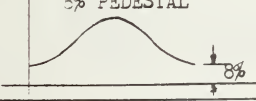
Figure 8 - Eanning Function Fourier Transform Pair.



It should be remembered that the non-zero frequency components are considerably BROADENED or SMEARED with respect to the impulse function. In general, the more the leakage is reduced, the broader or more smeared the results of the discrete Fourier transform appear. Table-II contains some of the more commonly used data windows.



TABLE-II  
Data Windows

WEIGHTING FUNCTION	3-dB BW	NOISE BW	HIGHEST SIDELOBE LEVEL (dB)	ASYMPTOTIC ROLL-OFF (dB/OCTAVE)
RECTANGULAR 	0.85B	1B	-13	6
TRIANGLE 	1.25B	1.35B	-26	12
COSINE 	1.17B	1.26B	-23	12
HANNING = (COSINE) <sup>2</sup> 	1.4B	1.5B	-32	18
(COSINE) <sup>3</sup> 	1.61B	1.73B	-39	24
(COSINE) <sup>4</sup> 	1.88B	1.9B	-48	30
HAMMING = (COSINE) <sup>2</sup> + 8% PEDESTAL 	1.3B	1.36B	-42	6dB/OCT BEYOND 5B





## D. SPECTRUM ANALYZER PERFORMANCE FIGURES OF MERIT

### 1. Sampling Frequency and Anti-Aliasing Filter

The sampling frequency must satisfy the Nyquist sampling theorem or aliasing will result. For data that has components beyond the analysis range, the components are attenuated by means of a lowpass filter, known as an "anti-aliasing" filter. A typical "fall off" rate for anti-aliasing filters in today's equipment is 120 dB/octave. In practice, a sampling rate of approximately three times the width of the desired analysis range is a good compromise. Sampling at higher rates requires more equipment, which is undesirable for economic reasons. Sampling at lower rates fails to take into account the actual fall-off of available anti-aliasing filters, and thus does not eliminate aliasing completely.

### 2. Bandwidth (Frequency Range)

The bandwidth determining parameters of a spectrum analyzer are:

1. The memory capacity (which determines the number of samples available for processing).
2. The sampling frequency (which, in combination with 1, determines the length of signal (in seconds) which is available for processing).
3. The shape of the "window function" also known as the "weighting function".



### 3. Resolution of 3-dB Bandwidth

In Table-II the 3-dB bandwidth for various weighting functions were compared using the relation

$$B(HZ) = \frac{1}{\text{Weighting-function length (sec)}}$$

The 3-dB bandwidth is a good measure of a spectrum analyzer's ability to resolve two equal-amplitude adjacent sine waves. Studying Table-II, it appears that the rectangular weighting function has the best resolution, but in reality, the response characteristic is limited in usefulness by the side-lobe structure. In practice, it is more important to resolve unequal than equal amplitude adjacent frequency components.

### 4. Noise Bandwidth

Sometimes called "effective bandwidth", it is used to convert or normalize power spectrum measurements to power spectral density (power per unit frequency). The noise bandwidth is the bandwidth of a hypothetical rectangular filter, which passes a signal with the same mean-square value as the actual filter when the filter input is white Gaussian noise.

### 5. Highest Sidelobe Level and Asymptotic Roll-off

It is useful to specify these parameter, since it provides some indication of the prominence of the side-lobe structure. The largest sidelobe is generally the first one,



with the Hamming function being among the exceptions. The third side lobe of the Hamming function is the largest.

The presence of side lobes makes it difficult to distinguish between a high-level frequency component and a low-level component which is close in frequency. For this reason, rectangular weighting, which leads to a spectrum response equal to the true Fourier transform of the time-truncated signal, is used only occasionally when analyzing transients or shocks.

## 6. Single-Sided Spectra

Although it is not an actual measure of performance, a discussion of single-sided spectra is warranted at this time so that the results obtained from the various algorithms is fully understood.

The finite discrete transform presumes that the input waveform is the sum of cosine-phase and sine-phase frequency components, all harmonically-related to the lowest frequency that can be recognized. That fundamental is the frequency that completes one cycle in the length of the input frame. The calculated harmonics range from (d-c) through 1 (the fundamental), 2, 3, etc., to  $N/2$ , the maximum recognizable harmonic, which completes one cycle in the two intervals associated with two points of the input (i.e., two samples per cycle).

In the general case, the discrete Fourier transform can operate on a complex time series input function and produce a complex spectrum. In such a case, an input function of  $N$  complex values (i.e.,  $2N$  independent real and imaginary parts) yields a spectrum of  $2N$  independent real and imaginary parts representing  $N$  complex frequency



coefficients centered about 0 (d-c) and ranging from harmonic  $-N/2$  to  $+N/2$ . The "negative-frequency" components are an integral and necessary part of the function.

A real input function, as an electrical waveform, may be considered for this general case as a complex function whose imaginary parts are all zero. The discrete transform of such an input yields a complex spectrum as above. For this special-case input however, the spectrum will be symmetrical about the center or d-c point. The negative-frequency parts from 0 to  $-N/2$  are an image of the positive frequencies, the real or cosine parts having even symmetry and the imaginary, sine parts odd. The negative-frequency half thus becomes redundant, and an input of  $N$  real points yields  $N$  independent real and imaginary parts of  $N/2$  complex spectral coefficients.

Nonetheless, the negative-frequency coefficients are still a necessary part of the spectrum in that they represent half the amplitude of the harmonic components. In other words, if the input function contains a cosine-phase component of amplitude  $A$  at frequency  $f_0$ , then the resultant positive-frequency coefficient for frequency  $f_0$  will have amplitude  $A/2$  and the negative-frequency coefficient for  $-f_0$  will also have amplitude  $A/2$ . The 0<sup>th</sup> or d-c coefficient, however, dividing or being "shared" between the positive and negative halves of the spectrum, will have full amplitude  $A$  for an input d-c level of  $A$ , which is the point of this discussion.





## E. SPECTRAL ANALYSIS ALGORITHMS

### 1. The Fast Fourier Transform

The fast Fourier transform (FFT) is a highly efficient procedure for computing the DFT of a time series. It takes advantage of the fact that the calculation of the coefficients of the DFT can be carried out iteratively, which results in a considerable savings of computation time. To gain insight to the FFT algorithm, a general development is presented followed by several examples. A more in depth study of the FFT may be found in Refs.[6 ], [16], and [17].

#### a. General Development of the FFT Algorithm

The discrete Fourier transform Eq.[1-3] is

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j(2\pi/N)nk} \quad , \quad k=0,1,\dots,N-1 \quad (1-3)$$

which may be written as

$$X(k) = \sum_{n=0}^{N-1} x(n) W^{nk} \quad (1-9)$$



where

$$W = e^{-j2\pi/N} \quad (1-10)$$

As can be seen,  $W^{nk}$  is periodic and of period  $N$ ; i.e.,

$$W^{(n+mN)(k+lN)} = W^{nk}, \quad m, l = 0, \pm 1, \dots \quad (1-11)$$

In carrying out the FFT algorithm, the symmetry and periodicity of  $W^{nk}$  are exploited to achieve an increase in efficiency. To emphasize this periodicity,  $W$  will be rewritten as  $W_N$  in carrying out the general development.

In Eq.[1-9],  $W^{nk}$  is a complex exponential, thus an examination of Eq.[1-9] shows that, in the case when  $x(n)$  is a complex sequence, a complete direct evaluation of an  $N$ -point DFT requires  $(N-1)^2$  complex multiplications and  $N(N-1)$  complex additions.<sup>1</sup> Fig.[9] shows the comparison of multiplications required by the direct calculation of the DFT and for the base 2 FFT algorithm.

<sup>1</sup>In most literature,  $N$  is assumed to be large, so that  $(N-1)$  can be approximated by  $N$ .



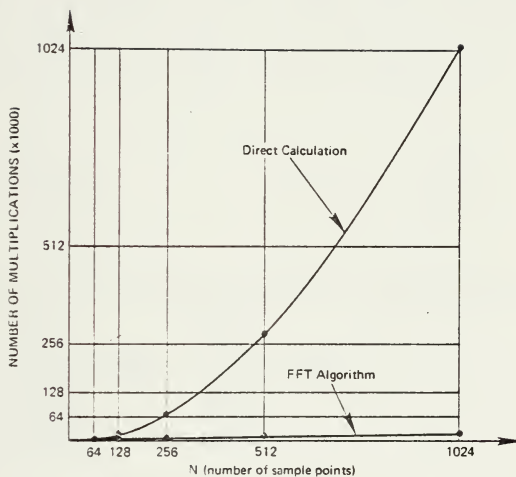


Figure 9 - Multiplications Required to Compute the DFT.



## k. Decimation-In-Time (DIT) Algorithm

The general principle behind the FFT is to break the original  $N$ -point sequence into two shorter sequences, the DFT's of which can be combined to give the DFT of the original  $N$ -point sequence.

Assuming  $N$  is a power of 2, define two  $(N/2)$ -point sequences  $x_1(n)$  and  $x_2(n)$  as the even and odd members of the sequence  $x(n)$  respectively, i.e.,

$$\begin{aligned} x_1(n) &= x(2n) & n &= 0, 1, \dots, \frac{N}{2} - 1 \\ x_2(n) &= x(2n+1) & n &= 0, 1, \dots, \frac{N}{2} - 1 \end{aligned} \quad (1-12)$$

The  $N$ -point DFT of the sequence  $x(n)$  can be written as

$$X(k) = \sum_{\substack{n=0 \\ n \text{ even}}}^{N-1} x(n) w_N^{nk} + \sum_{\substack{n=0 \\ n \text{ odd}}}^{N-1} x(n) w_N^{nk} \quad (1-13)$$

$$= \sum_{n=0}^{N/2-1} x(2n) w_N^{2nk} + \sum_{n=0}^{N/2-1} x(2n+1) w_N^{(2n+1)k} \quad (1-14)$$

Then using the following identity

$$w_N^2 = \left[ e^{j(2\pi/N)} \right]^2 = e^{j[2\pi/(N/2)]} = w_{N/2} \quad (1-15)$$

Eq. [1-14] can be written in the form

$$X(k) = \sum_{n=0}^{N/2-1} x_1(n) w_{N/2}^{nk} + \sum_{n=0}^{N/2-1} x_2(n) w_{N/2}^{nk} \quad (1-16)$$





$$X(k) = X_1(k) + W_N^k X_2(k) \quad (1-17)$$

Each of the sums in Eq.[1-16] is an  $N/2$ -point DFT, the first sum being the  $N/2$  point DFT of the even-numbered points of  $x(n)$  and the second being the  $N/2$  point DFT of the odd-numbered points of the original sequence. Although the index  $k$  ranges over  $N$  values,  $k=0,1,\dots,N-1$ , each of the sums need only be computed for  $k$  between 0 and  $N/2-1$ , since  $X_1(k)$  and  $X_2(k)$  are each periodic in  $k$  with period  $N/2$ . After the two DFT's corresponding to the two sums in Eq.[1-16] are computed, they are then combined to yield the  $N$ -point DFT,  $X(k)$  as given by Eq.[1-17].



## 2. The Chirp-Z-Transform (CZT)

The chirp-z-transform (CZT) is an algorithm for evaluating the z-transform of a finite duration sequence along certain general contours in the z-plane. The following derivation is based upon a report by Rabiner, Schafer, and Eader [18].

### a. The Derivation of the CZT Algorithm

In general, the z-transform of a sequence of samples  $\{x(n)\}$  is defined as

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) z^{-n} \quad (1-18)$$

Assuming that the right side of Eq.[1-18] converges for some values of  $z$ , and restricting the evaluation of the z-transform to a finite duration sequence of samples, Eq.[1-18].

$$X(z) = \sum_{n=0}^{N-1} x(n) z^{-n} \quad (1-19)$$

The special case of the z-transform which has received considerable attention is the set of points equally spaced around the unit circle of the z-plane, i.e.,



$$Z_k = \exp(j2\pi k/N), \quad k=0,1,\dots,N-1 \quad (1-20)$$

which gives

$$X(z) = \sum_{n=0}^{N-1} x(n) e^{-j(2\pi/N)nk}, \quad k=0,1,\dots,N-1 \quad (1-21)$$

By comparing Eq.[1-21] and Eq.[1-3], one can see that it is the discrete Fourier transform (DFT). The DFT can be evaluated more efficiently by using the FFT rather than the CZT, but the latter allows one to evaluate Eq.[1-19] along the more general contour

$$Z_k = A W^{-k}, \quad k=0,1,\dots,M-1 \quad (1-22)$$

where  $M$  is an arbitrary integer (not necessarily equal to  $N$ ) and  $A$  and  $W$  are arbitrary complex numbers defined by

$$A = A_o \exp(j2\pi\theta_o) \quad (1-23)$$

$$W = W_o \exp(j2\pi\phi_o) \quad (1-24)$$

Fig.[10] shows the significance of  $A_o$ ,  $W_o$ ,  $\theta_o$ , and  $\phi_o$  in the  $z$ -plane.



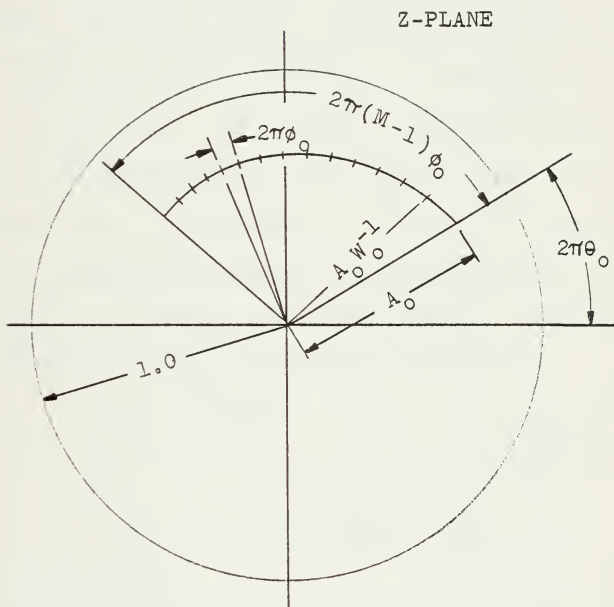


Figure 10 - Contour of the CZT in the Z-Plane.





For the special case of the DFT,  $A=1$ ,  $M=N$ , and  $W=\exp(-j2\pi/N)$ ; thus the DFT coefficients of a finite duration sequence are the values of the z-transform of that same sequence at  $N$  evenly spaced points around the unit circle.

The parameter  $W_0$  determines the rate at which the contour spirals: if  $W_0$  is greater than unity, the contour spirals toward the origin as  $K$  increases, and if  $W_0$  is less than unity, the contour spirals outward as  $K$  increases. The parameters  $A_0$  and  $2\pi\theta_0$  are the location in radius and angle, respectively, of the first sample, i.e., for  $k=0$ . The remaining samples are located along the spiral contour with an angular spacing of  $2\pi\phi_0$ . The equivalent s-plane contour of Fig.[10] is shown in Fig.[11].

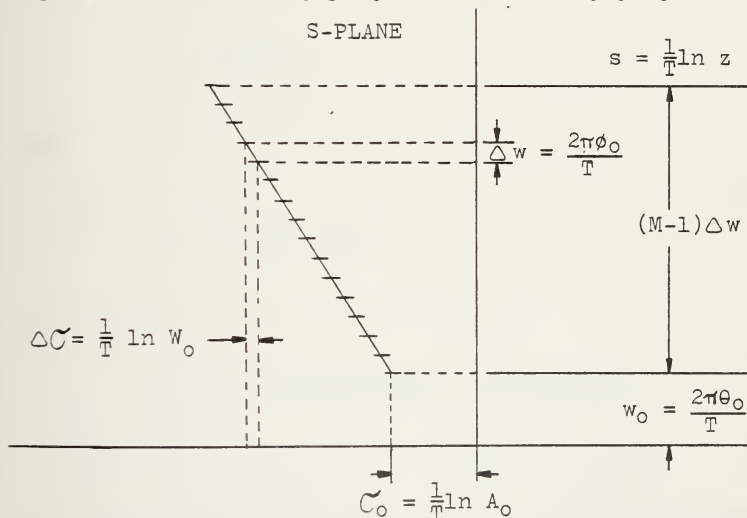


Figure 11 - Contour of the CZT in the S-Plane.



The equivalent s-plane contour beginning point may be found by making the substitution

$$z = e^{sT}$$

then

$$z_c = A = \exp(s_o T)$$

or

$$s_c = (1/T) \ln A$$

$$= \sigma_c + j\omega_o = (1/T) (\ln A_o + j2\pi\theta_o) \quad (1-25)$$

For a general point on the s-plane contour

$$z_k = \exp(s_k T) = AW^{-k}$$

then

$$s_k = (1/T) \ln(AW^{-k})$$

and finally

$$s_k = (1/T) (\ln A - k \ln W) = s_c - (k/T) \ln W$$

$$k=0, 1, \dots, M-1 \quad (1-26)$$

Thus the spiraling contours in the z-plane correspond to straight lines in the s-plane. The rate of spiraling in the



z-plane determines the slope of the line in the s-plane.

The task now at hand is to compute Eq.[1-19] along a general contour given by Eq.[1-22]. The evaluation of  $X(z)$  at  $z=z_k$ , will be written using the notation  $X_k$ , i.e.,

$$X_k = \sum_{n=0}^{N-1} x(n) z_k^{-n} \quad (1-27)$$

or

$$X_k = \sum_{n=0}^{N-1} x(n) A^{-n} W^{nk}, \quad k=0, 1, \dots, M-1 \quad (1-28)$$

where  $N$  is the length of the sequence  $x(n)$ .

Making the following substitution, which was originated by Elvestein [5].

$$nk = \frac{n^2 + k^2 - (k-n)^2}{2} \quad (1-29)$$

gives



$$X_k = \sum_{n=0}^{N-1} x(n) A^{-n} W^{[n^2+k^2-(k-n)]/2}$$

$$= \sum_{n=0}^{N-1} [x(n) A^{-n} W^{n^2/2}] W^{k^2/2} W^{-(k-n)^2/2} \quad (1-30)$$

Although Eq.[1-30] appears to be more complicated than Eq.[1-28], it allows efficient hardware implementation. Eq.[1-30] may be viewed as basically a three step process:

1. The first step consists of a complex pre-multiplication forming a new sequence

$$y(n) = x(n) A^{-n} W^{n^2/2}, n=0,1,\dots,N-1 \quad (1-31)$$

2. The sequence  $y(n)$  is then convolved with the sequence  $v(n)$  defined as

$$v(n) = W^{-n^2/2} \quad (1-32)$$

to give the sequence  $g(k)$  defined as

$$g(k) = \sum_{n=0}^{N-1} y(n) v(k-n), k=0,1,\dots,M-1 \quad (1-33)$$

3. The sequence  $g(k)$  is then post-multiplied by  $W^{k^2/2}$  to give





$$X_k = W^{k^2/2} \sum_{n=0}^{N-1} y(n) v(k-n), \quad k=0, 1, \dots, M-1 \quad (1-34)$$

A pictorial representation of the processing operations of the CZT algorithm is shown in Fig.[12].

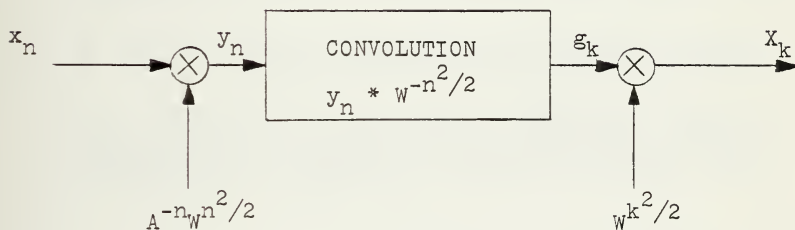


Figure 12 - Processing Operations of the CZT Algorithm.



The advantages of the CZT algorithm over the standard FFT are summarized below:

1.  $N$ , the number of points in the input sequence, doesn't have to be equal to  $M$ , the number of points at which the  $z$ -transform is evaluated.

2. Neither  $N$  nor  $M$  has to be composite - both can be prime numbers.

3. The starting point in the  $z$ -plane and the angular spacing of the  $z^k$ 's are arbitrary. Thus the frequency resolution and the range of frequencies are arbitrary.

4. The contour does not have to be a circle in the  $z$ -plane. The CZT algorithm can evaluate the  $z$ -transform along spiral contours inside and outside the unit circle of the  $z$ -plane. By evaluating the  $z$ -transform along a selected contour, the transfer function of a linear system can be adjusted to sharpen or flatten the frequency response. The choice of contour is dependent on which of the system's poles should be emphasized.

5. If  $A=1$ ,  $M=N$ ,  $W = \exp(-j2\pi/N)$ , then the CZT can be used to evaluate the DFT, even when  $N$  is prime.

6. With the CZT algorithm, interpolation between time samples of a bandlimited function can be obtained with greater ease with the DFT, and interpolation at arbitrary sampling intervals can be performed efficiently.



### 3. The Prime Transform

#### a. The Prime Transform Derivation

The following derivation is based on a paper by Whitehouse, Means, and Speiser [26]. A brief review of the number theory properties used in this derivation is given in Appendix B.

In the specific case when  $N$  is an odd prime, Eq.[1-3] can be written as

$$X(C) = \sum_{n=0}^{N-1} x(n) \quad (1-35)$$

to give the (D-C) frequency component and

$$X(k) - x(0) = \sum_{n=1}^{N-1} x(n) e^{-j2\pi nk/N}, k=1, \dots, N-1 \quad (1-36)$$

to give the remaining components.

Since only non-zero values of  $n$  and  $k$  occur in the right hand side of Eq.[1-36], it is possible to replace the product  $nk$  by a primitive root raised to a sum [19].

Proceeding, in the summation of Eq.[1-36], the terms are permuted and the order of the equation changed by



the following transformations:

$$\begin{aligned} (n)_p \rightarrow ((r^{(n)}_p)) &= r^n \text{ modulo } N \\ (k)_p \rightarrow ((r^{(k)}_p)) &= r^k \text{ modulo } N \end{aligned} \quad (1-37)$$

where "r" is a primitive root and "p" implies permuted.  
 Noting that  $r^{N-1} \text{ modulo } N = r^0 \text{ modulo } N$ , Eq.[1-36] is written as

$$\left\{ X_{((r^{(k)}_p))} - x(0) \right\} = \sum_{n=0}^{N-1} x(n)_{((r^{(n)}_p))} \exp(-j \frac{2\pi}{N} r^{((n)_p + (k)_p)}) \quad (1-38)$$

An examination of Eq.[1-38], shows that the sequence  $\left\{ X_{((r^{(k)}_p))} - x(0) \right\}$  is the circular correlation of the sequence  $\left\{ \exp -j(2\pi/N) r^{(n)_p} \right\}$ .

In matrix notation, Eq.[1-36] may be written as

$$\mathbf{x}' - x(0)\mathbf{D} = \mathbf{F}' \mathbf{x}'(n) \quad (1-39)$$

where

$$\mathbf{D} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \quad (1-40)$$





and  $X'$  and  $x'$  (n) are column vectors of size (N-1) derived from  $X(k)$  and  $x(n)$  by deleting, respectively,  $X(0)$  and  $x(0)$ .

The matrix  $F'$  can be factored into three matrices

$$F' = P^t C F \quad (1-41)$$

where  $F$  is an (N-1) permutation matrix,  $C$  is an [(N-1) x (N-1)] circulant matrix, and  $P^t$  is the transpose of  $P$ .

The elements of the  $C$  matrix are

$$C_{r,k} = F'^{(n)p,(k)p} \quad , \quad n,k=1,\dots,N-1 \quad (1-42)$$

where "r" is a primitive root of  $N$  and "p" implies permuted. The elements of the permutation matrix,  $P$ , are

$$P_{n,k} = \delta_{(n)p,k} \quad , \quad n,k=1,\dots,N-1 \quad (1-43)$$

where  $\delta_{(n)p,k}$  is the Kronecher delta function. The circulant matrix,  $C$ , can be implemented with a recirculating transversal filter whose tap weights are determined by

$$h(r) = W^{(n)p} \quad , \quad n=1,\dots,N-1 \quad (1-44)$$



Thus the prime transform algorithm offers more simplification than the CZT algorithm because in hardware implementation it is possible to eliminate the pre- and post-multipliers. The processing operations may be summarized as

1. permutation of the input data
2. circular correlation
3. permutation of the output Fourier coefficients



## F. HARDWARE AND SOFTWARE IMPLEMENTATION

in both hardware and software implementation of these algorithms, the goal is to have the ability to do real-time signal processing for the frequency range of interest. In spectral analysis applications, signal processing takes place in real time when the spectrum level of each frequency component is updated at a rate equal to at least the bandwidth  $\beta$  (Hz) of the system.

Depending upon the system, the signal for processing may be in any of three forms. i.e.,

1. Analog  $\rightarrow$  both time and signal amplitude are continuous
2. Sampled analog  $\rightarrow$  time is discrete and signal amplitude is continuous.
3. Digital  $\rightarrow$  time and signal amplitude are both discrete

### 1. Analco Spectral Analysis

#### a. Scanning Analyzers

The conventional scanning-filter analyzer consists of a single bandpass filter, whose center frequency is "shifted" by using frequency-translation techniques. Typically a single bandpass filter is employed and the filter analyzes one frequency at a time.



Since a single filter is employed and is applied successively to each frequency, it must dwell for a period of  $1/\beta$  at each frequency. Thus  $(1/\beta) \times N_c$  of steps equals the total analyzing time "T". The narrower the filter the longer is the total sweep time. Since each frequency component is updated only once per time T, real time signal processing is not possible.

## b. Multifilter Spectrum Analyzers

Constant bandwidth multifilter analyzers were developed for real time analysis. In this type of analyzer, the signal passes through many parallel filters simultaneously, typically 500 contiguous magnetostrictive filters per analysis range. The filters and the associated circuits must possess uniform and constant gain so that a fixed amplitude sine wave of any frequency in the analysis range results in a spectrum output which is constant. However, frequency flatness of better than  $\pm 3\text{dB}$  is difficult to maintain, thus the magnetostrictive multifilter analyzers are not used extensively.

## 2. Digital Spectral Analysis

The all digital spectrum analyzers employ a computer or computer-like circuitry which is programmed to implement the FFT algorithm. Some of the structural factors for the design of such a system are discussed below.





TABLE-III  
Comparison of Major Logic Families

	RTL	DTL	TTL	ECL	P-MOS	CMOS
Typical Output Impedance	Kilohms	Kilohms	10-17 ohms	6-15 ohms	Kilohms	Kilohms
Fan-out	5	8	10	10-25	20	50
Typical Power Dissipation per Gate	2.5-12 mW	8-12 mW	12-22 mW	40-55 mW	0.2-10 mW	1 mW (at 1 MHz)
Immunity to External Noise	fair	good	very good	good	fair	very good
Noise Generation	low-medium	medium	medium-high	low-medium	medium	low-medium
Propagation Delay per Gate (n sec)	12-25	30	6-12	1-4	300	70
Typical Clock Rate for Flip-Flop MHz	2.5-8	12-30	15-60	60-400	2	5
Versatility	good	fair	very good	good	low but growing	low but growing

Taken from Ref.[17].



## a. Technology

Basic speed and functional blocks available are dictated by technology. Table-III shows a comparison of the major logic families. Because of speed-complexity tradeoffs, the choice of logic family is usually not a clear-cut decision. This complexity encompasses many areas, but important are available MSI and LSI functional modules, system interconnections, multiple function modules, and type of multiplier. To gain speed in the FFT algorithm a fast multiplier is required in the butterflies, thus the type of multiplier is an important consideration. Table-IV is a comparison of multipliers.

TABLE-IV  
MULTIPLIER COMPARISON TABLE

Configuration	Type of Basic Package	Component Technology	Computation Time (nsec)			Package Count		
			16 × 12	16 × 8	9 × 9	16 × 12	16 × 8	9 × 9
Fast trapezoid (Fig. 8.35)	Custom two-bit adder package	ECL	32	28	22	112	82	45
Tree	Commercial four-bit adder package	ECL	62	48	40	160	120	85
Tree	Commercial four-bit adder package	TTL	155	100	85	160	120	85
	(2 × 4)-bit (commercial) array package	TTL	261	195	135	80	58	30
Two bits at a time add-shift	Custom two-bit adder package	ECL	130	120	100	40	36	25
Two bits at a time add-shift	Commercial two-bit adder package	ECL	210	140	116	40	36	25
Two bits at a time add-shift	Commercial two-bit adder package	TTL	450	300	250	40	36	25

Taken from Ref. [17].



## b. Algorithm

The structure of the FFT algorithm is a major factor in determining whether or not special-purpose hardware or a general-purpose computer is used in a particular application. Table-V lists the arithmetic operations required for different radix algorithms. When implemented with software, the radix-8 algorithm is near optimum. However, it lacks the flexibility required of a general-purpose computer and thus is only used in special purpose applications. Although the radix-2 requires additional computations, its simplicity has made it popular for use in general purpose computers where real time analysis is not a requirement.

## c. Hardware Architecture

When implemented with hardware, the radix-4 algorithm has received the most attention. Hardware implementation of the FFT algorithm has gained speed by trading off flexibility. The speed is gained by a complex mix of several types of parallelism. One such type is pipelining, i.e., the process is broken up into several sequential tasks and execution proceeds assembly-line style. By pipelining many processing steps can be progressed at any given time and thus performing real time analysis. Given that an  $N$ -point FFT requires  $(N/2) \log_2 N$  butterflies, there are four basic ways to organize the processor.



TABLE-V  
Required Arithmetic Operations

Algorithm	Required Computation for	Real Multiplications	Real Additions
Base 2 algorithm for $N = 2^r$ $r = 0, 1, 2, \dots$	Evaluating $(N/2)r$ , 2 term Fourier transforms.	0	$2Nr$
	Referencing	$(r/2 - 1)N + 1(4)$	$(r/2 - 1)N + 1(2)$
	Complete analysis	$(2r - 4)N + 4$	$(3r - 2)N + 2$
Base 4 algorithm for $N = (2^r)^{1/2}$ $r/2 = 0, 1, 2, \dots$	Evaluating $(N/4)(r/2)$ , 4 term Fourier transforms.	0	$2Nr$
	Referencing	$(3r/8 - 1)N + 1(4)$	$(3r/8 - 1)N + 1(2)$
	Complete analysis	$(1.5r - 4)N + 4$	$(2.75r - 2)N + 2$
Base 8 algorithm for $N = (2^r)^{1/3}$ $r/3 = 0, 1, 2, \dots$	Evaluating $(N/8)(r/3)$ , 8 term Fourier transforms.	$Nr/6$	$13Nr/6$
	Referencing	$(7r/24 - 1)N + 1(4)$	$(7r/24 - 1)N + 1(2)$
	Complete analysis	$(1.333r - 4)N + 4$	$(2.75r - 2)N + 2$
Base 16 algorithm for $N = (2^r)^{1/4}$ $r/4 = 0, 1, 2, \dots$	Evaluating $(N/16)(r/4)$ , 16 term Fourier transforms.	$3Nr/8$	$9Nr/4$
	Referencing	$(15r/64 - 1)N + 1(4)$	$(15r/64 - 1)N + 1(2)$
	Complete analysis	$(1.3125r - 4)N + 4$	$(2.71875r - 2)N + 2$

Taken from Ref. [6].





1. Sequential→ all  $(N/2) \log_2 N$  butterflies are performed sequentially.
2. Cascade→ if  $\log_2 N$  arithmetic units (AU) are available, the algorithm can be pipelined so that each AU computes  $(N/2)$  butterflies before advancing data through the pipe.
3. Parallel-Iterative→ if  $(N/2)$  AU's are available, the algorithm can be processed in a  $\log_2 N$  stage pipeline. This requires  $\log_2 N$  butterfly times.
4. Array→ if  $(N/2) \log_2 N$  AU's are available, the entire array can be pipelined into one execution time.

Table-VI compares these different organizations for an  $N=1024$  FFT and an execution time of  $1 \mu s$ . The total execution time does not take into account the I/O overhead time.

TABLE-VI  
Comparison of Processor Organizations

Machine Organization	Arith. Units (Butterflies)	Total Execution Time ( $\mu s$ )
Sequential	1.	5120.
Cascade	10.	512.
Parallel Iterative	512.	10.
Array	5120.	1.



In addition to pipelining, among the types of parallelism are

1. Time overlap of control and memory functions.
2. The addition of a small amount of high speed memory.
3. Higher radix algorithms.

Table-VII compares some important parameters of a few modern signal processing systems.

### 3. Sampled Analog Spectral Analysis

Like the FFT algorithm the CZT algorithm can compute the frequency components of a signal more efficiently than by direct evaluation of the DFT. However, for relatively large values of  $N$ , the CZT algorithm is less efficient when compared to the FFT algorithm. For this reason interest to digitally implement the CZT algorithm waned while great strides were being taken to develop FFT software and hardware.

The CZT does offer many signal processing advantages, but only if it can be implemented using sampled analog techniques. With the development of Surface Acoustic wave (SAW) and Charge Transfer Devices (CTD), interest in the CZT has been re-kindled. The prime advantage of these devices over digital equipment is that when the continuous analog signal is sampled the analog value is retained, while in a digital system the analog value must be quantized which is a source of error.

CTD's, which include the Charge Coupled Device (CCD) and Bucket-Brigade Device (BBD), sample the optical or



electrical input and convert it to a packet of charge whose magnitude is proportional to the input. The charge packet moves down the CTD stages, shift register fashion, under the control of clock voltages. The charge packets are transferred from one stage to the next until they reach an output stage where the magnitude of the charge packet is sensed and converted to an electrical analog signal. The clock frequency and the number of stages transferred determine the time delay introduced.

In a SAW device a mechanical ripple, or wave, travels along a solid polished surface in much the same manner as an ocean wave travels through water. Aluminum electrodes are usually deposited on a piezoelectric material such as ST quartz, and they act to launch and receive the acoustic waves. These transducers convert energy between electrical and mechanical domains.

In a CCD the delay line is tapped and weighted by using a split-gate weighting technique which is incorporated into the mask at the time of fabrication or by weighting the taps with external resistors. The tap weight in a SAW device is photo-lithographically determined and is proportional to the length of the interdigitations which accept the acoustic wave. The pre-and post-multipliers are accomplished using wideband four analog multipliers with weights generated by programmable read only memories (PROM) and coupled through D/A's or by multiplying D/A's with PROM weight storage.

The transversal filter implementation of the CZT algorithm achieves its high speed not by reducing the number of multipliers like the FFT, but by putting the equivalent of  $N$  parallel multipliers all on the same chip. This reduces the number of computational steps from  $N \log_2 N$  for the FFT to



N for the CZT. Also the N additions per component are simultaneous. Currently, the CCD CZT can operate in real time at speeds up to 5MHz and the EBD CZT is limited to a sample rate of a few hundred KHz. For higher sample rates, up to several MHz, SAW devices are attractive. In addition, the sampled analog devices require low power and are light weight. The advantages of the digital FFT are its flexibility and accuracy. Programability is being developed for the CCD and SAW filters with up to 128 digitally switched taps have been built.

As noted earlier, the prime transform offers additional simplicity by eliminating the multipliers required in the CZT implementation. The multipliers are replaced with analog permuter memories. The commercially available serial access memory stores analog samples as charges in an array of MOS capacitors under the control of read in and read out shift registers. Currently the maximum sample rate for this type of device is 5 MHz.





TABLE-VII

## Comparison of Signal Processors

	Small-Signal Processor	Lin. Labs. Digital - Voice Term- inal	Lin. Labs. Fast Digital Processor	Signal-Pro- cessing Systems	TI's Advanced Scientific Computer
Logic Family	ECL	ECL	ECL	TTL	ECL
Number of In- tegrated Cir- cuits	500	470	10,000	1400	500,000 Equivalent Gates
Multiply time	150 ns	212 ns	450 ns	1 us	75 ns
Basic Cycle Time	100 ns	53 ns	150 ns	200 ns	75 ns
Word Length	24 bits	16 bits	18 bits	16 bits	32 bits
FFT Time, 1024 Point Complex	5.5 ms	5.0 ms	5.5 ms	8.3 ms	8.0 ms

Taken from Ref. [1].



## II. FAST FOURIER TRANSFORM ALGORITHMS

### A. MATHEMATICAL ILLUSTRATION OF THE DECIMATION-IN-TIME (DIT) ALGORITHM

To illustrate the FFT algorithm, the number of sample points of  $x(n)$  is assumed to be a power of 2, i.e.,  $N = 2^Y$ . In this example  $N=4=2^2$  and  $Y=2$ .

For  $N=4$ , Eq.[1-3] can be written as

$$\begin{aligned} X(0) &= x(0)W^0 + x(1)W^0 + x(2)W^0 + x(3)W^0 \\ X(1) &= x(0)W^0 + x(1)W^1 + x(2)W^2 + x(3)W^3 \\ X(2) &= x(0)W^0 + x(1)W^2 + x(2)W^4 + x(3)W^6 \\ X(3) &= x(0)W^0 + x(1)W^3 + x(2)W^6 + x(3)W^9 \end{aligned} \quad (2-1)$$

or written in matrix form

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & W^0 & W^0 \\ W^0 & W^1 & W^2 & W^3 \\ W^0 & W^2 & W^4 & W^6 \\ W^0 & W^3 & W^6 & W^9 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} \quad (2-2)$$



The first step in developing the FFT algorithm for this example is to rewrite Eq.[2-2] using the following relation from number theory[23].

$$W^{rk} = W^{rk \bmod(N)} \quad (2-3)$$

where  $[nk \bmod(N)]$  is the remainder upon division of  $nk$  by  $N$ . Hence if  $N=4$ ,  $r=3$ ,  $k=3$

$$W^9 = W^1 \quad (2-4)$$

since

$$\begin{aligned} W^{nk} &= W^9 = \exp[(-j2\pi/4)(9)] = \exp(-j9\pi/2) \\ &= \exp(-j\pi/2) = \exp[(-j2\pi/4)(1)] \end{aligned}$$

$$= W^1 = W^{rk \bmod(N)} \quad (2-5)$$

Thus the new matrix is written as

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & W^1 & W^2 & W^3 \\ 1 & W^2 & W^0 & W^2 \\ 1 & W^3 & W^2 & W^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} \quad (2-6)$$

The second step is the factorization of the square matrix in Eq.[2-6]. As an aid in showing how Eq.[2-6] is factored, a new notation is introduced at this time, where the integers  $n$  and  $k$  are now represented as binary numbers.



$$n = 0, 1, 2, 3 \rightarrow n(n_1, n_0) = 00, 01, 10, 11$$

$$k = 0, 1, 2, 3 \rightarrow k(k_1, k_0) = 00, 01, 10, 11$$

Then writing the base 10 equivalents

$$n = 2n_1 + n_0, \quad k = 2k_1 + k_0 \quad (2-7)$$

Eq.[1-9] can now be written as

$$X(k_1, k_0) = \sum_{n_0=0}^1 \sum_{n_1=0}^1 x(n_1, n_0) w^{(2n_1+n_0)(2k_1+k_0)} \quad (2-8)$$

The important point here, is that the single summation of Eq.[1-9] must be replaced by " " summations in order to enumerate all the bits of the binary representation of n.

Continuing

$$\begin{aligned} w^{(2n_1+n_0)(2k_1+k_0)} &= w^{(2k_1+k_0)2n_1} w^{(2k_1+k_0)n_0} \\ &= \left[ w^{4n_1k_1} \right] w^{2k_0n_1} w^{(2k_1+k_0)n_0} \\ &= w^{2k_0n_1} w^{(2k_1+k_0)n_0} \end{aligned} \quad (2-9)$$

The term in brackets is equal to unity since

$$w^{4n_1k_1} = \left[ w^4 \right]^{n_1k_1} = \left[ e^{-j2\pi 4/4} \right]^{n_1k_1} = \left[ 1 \right]^{n_1k_1} = 1 \quad (2-10)$$





Eq.[2-8] can now be written as

$$X(k_1, k_0) = \sum_{n_0=0}^1 \left[ \sum_{n_1=0}^1 x(n_1, n_0) w^{2k_0 n_1} \right] w^{(2k_1 + k_0)n_0} \quad (2-11)$$

Each of the summations are treated individually and lead directly to the correct factorization of Eq.[2-6]. Rewriting the bracketed summation

$$x_1(k_0, n_0) = \sum_{n_1=0}^1 x(n_1, n_0) w^{2k_0 n_1} \quad (2-12)$$

Directly evaluating this summation gives

$$\begin{aligned} x_1(0,0) &= x(0,0) + x(1,0)w^0 \\ x_1(0,1) &= x(0,1) + x(1,1)w^0 \\ x_1(1,0) &= x(0,0) + x(1,0)w^2 \\ x_1(1,1) &= x(0,1) + x(1,1)w^2 \end{aligned} \quad (2-13)$$

re-writing in matrix notation

$$\begin{bmatrix} x_1(0,0) \\ x_1(0,1) \\ x_1(1,0) \\ x_1(1,1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & w^0 & 0 \\ 0 & 1 & 0 & w^0 \\ 1 & 0 & w^2 & 0 \\ 0 & 1 & 0 & w^2 \end{bmatrix} \begin{bmatrix} x(0,0) \\ x(0,1) \\ x(1,0) \\ x(1,1) \end{bmatrix} \quad (2-14)$$



Now writing the outer summation as

$$x_2(k_0, k_1) = \sum_{n_0=0}^1 x_1(k_0, n_0) w^{(2k_1+k_0)n_0} \quad (2-15)$$

Directly evaluating the summation gives

$$\begin{aligned} x_2(0,0) &= x_1(0,0) + x_1(0,1)w^0 \\ x_2(0,1) &= x_1(0,0) + x_1(0,1)w^2 \\ x_2(1,0) &= x_1(1,0) + x_1(1,1)w^1 \\ x_2(1,1) &= x_1(1,0) + x_1(1,1)w^3 \end{aligned} \quad (2-16)$$

Then in matrix notation

$$\begin{bmatrix} x_2(0,0) \\ x_2(0,1) \\ x_2(1,0) \\ x_2(1,1) \end{bmatrix} = \begin{bmatrix} 1 & w^0 & 0 & 0 \\ 1 & w^2 & 0 & 0 \\ 0 & 0 & 1 & w^1 \\ 0 & 0 & 1 & w^3 \end{bmatrix} \begin{bmatrix} x_1(0,0) \\ x_1(0,1) \\ x_1(1,0) \\ x_1(1,1) \end{bmatrix} \quad (2-17)$$

Combining Eqs. [2-12] and [2-15] results in

$$X(k_1, k_0) = x_2(k_0, k_1) \quad (2-18)$$

or writing in matrix form

$$\begin{bmatrix} X(0) \\ X(2) \\ X(1) \\ X(3) \end{bmatrix} = \begin{bmatrix} 1 & w^0 & 0 & 0 \\ 1 & w^2 & 0 & 0 \\ 0 & 0 & 1 & w^1 \\ 0 & 0 & 1 & w^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & w^0 & 0 \\ 0 & 1 & 0 & w^0 \\ 1 & 0 & w^2 & 0 \\ 0 & 1 & 0 & w^2 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} \quad (2-19)$$



A signal flow graph of Eq.[2-19] is shown in Fig.[13].

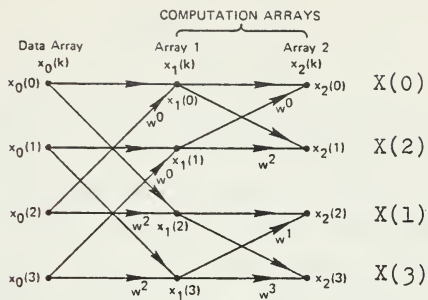


Figure 13 - FFT Signal Flowgraph,  $N=4$ .

Note that the final results are in bit reversed order with respect to the desired values  $X(k_1, k_0)$ . This is simply the scrambling which results from the FFT algorithm. The results are easily unscrambled by bit reversing, i.e.,

$X(00)$	$X(00)$	
$X(10)$	flips or reverses to	$X(01)$
		(2-20)
$X(01)$		$X(10)$
$X(11)$		$X(11)$

The algorithm just described is known as the Cooley-Tukey Algorithm or decimation-in-time (DIT) algorithm, since at each stage of the process the input sequence (i.e., time sequence) is divided into smaller sequences for processing.

The basic building block of the algorithm is the "butterfly" as shown in Fig.[14].



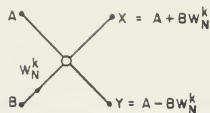


Figure 14 - Decimation-In-Time Butterfly.

Each butterfly requires one complex multiply (four real multipliers) and two complex adds. The importance of the butterfly structure to the FFT flow graph is that only one additional memory location is required to transform an  $N$ -point sequence stored in memory. Thus the intermediate results of the FFT are stored in the same locations in which the original data being transformed was stored. Since this algorithm stores both the input and output sequences in the same location, it is called an in-place algorithm.

As pointed out earlier the results of the DIT algorithm are bit reversed. This means that to get a normal ordered output, the input data must first be shuffled. Figs. [15a] and [15b] present the canonic signal flow graphs for the DIT algorithm for the case  $N=8$ . Note that the data is naturally ordered in Fig. [15a] and is in inverse order in Fig. [15b].





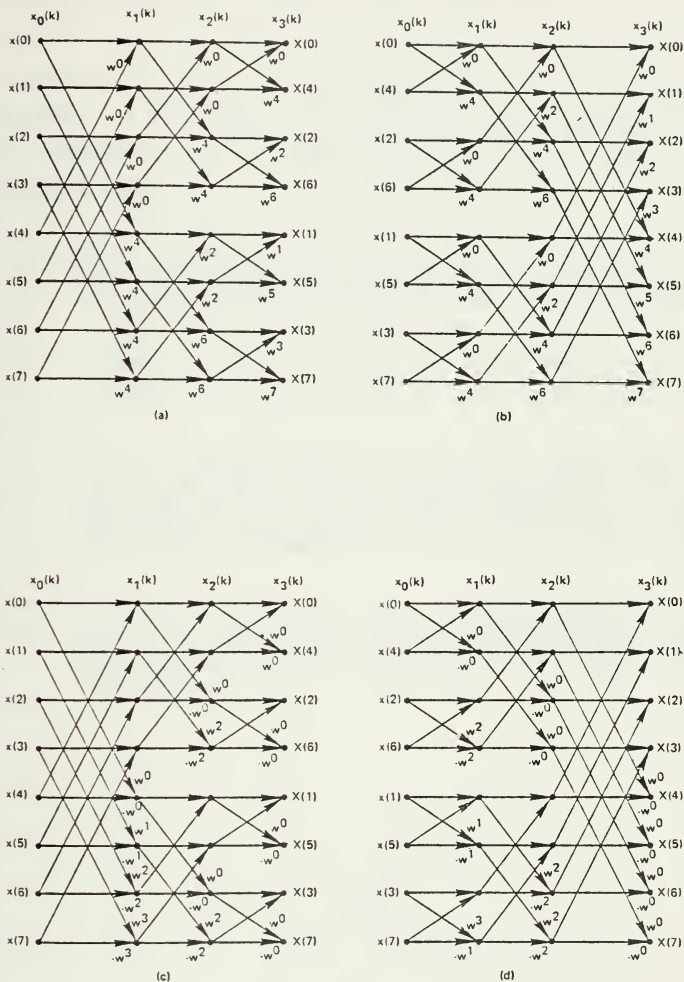


Figure 15 - FFT Canonic Flow Graphs.



## B. DECIMATION-IN-FREQUENCY (DIF) ALGORITHM

Another popular FFT algorithm is the Sande-Tukey Algorithm or decimation-in-frequency (DIF) algorithm. It is very similar to the DIT algorithm but differs in the following ways.

1. The input sequence  $\{x(n)\}$  is partitioned into two sequences each of length  $(N/2)$  samples in the following manner. The first sequence  $\{x_1(n)\}$  consists of the first  $(N/2)$  points of  $\{x(n)\}$ , whereas the second sequence  $\{x_2(n)\}$  consists of the last  $(N/2)$  points of  $\{x(n)\}$ .
2. For a given case in the DIT algorithm the input is bit-reversed while the output is in natural order, whereas the reverse is true for the DIF algorithm. In general, however, both the DIT and DIF can go from normal to shuffled data and vice versa.
3. The DIF butterfly is slightly different as shown in Fig.[16]. The difference being that the complex multiplication takes place after the add-subtract operation in the DIF algorithm.

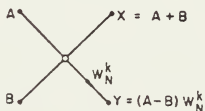


Figure 16 - Decimation-In-Frequency Butterfly.



Taking a look at the similarities, both algorithms require on the order of  $(N \log_2 N)$  operations to compute the DFT. Both algorithms can be done in-place and both need to perform bit reversal at some place during the computation.

Figs.[15c] and [15d] present the canonic signal flow graphs for the DIF algorithm for the case  $N=8$ . The data is naturally ordered in Fig.[15c] and is in inverse order in Fig.[15d]. The two most effective methods are those illustrated in Figs.[15b] and [15c] since they provide the powers of  $W$  in the correct order needed for computation. This eliminates the need for storage tables.

The FFT algorithms examined above by no means exhaust all the possibilities. Numerous algorithms have been developed, all with one basic purpose in mind, and that is to reduce the computation time.



### C. EXAMPLES USING THE FFT ALGORITHM

As a test of the FFT algorithm to compute the DFT, CASES 1, 2, and 3, as described in Appendix A, were used as inputs for  $N=32$ . For the LIT algorithm used,  $N$  is required to be highly composite and a power of 2. The value of 32 was used so that the results of this algorithm could be compared to the results of the CZT and prime transform algorithms. In the latter two algorithms  $N=31$ . All figures are drawn with normalized frequency, but for the sake of discussion, assume that the input sinusoids have frequency in Hertz and the rectangular data window " $T$ " is equal to 1 second.

The fundamental frequency is

$$F_c = 1/T = 1 \text{ Hz}$$

which is the minimum recognizable frequency component or resolution. the analyzing bandwidth is

$$F_{\max} = (N/2) (F_c) = 16 \text{ Hz}$$

The output is a one-sided spectra consisting of  $[(N/2)+1]$  frequency components. Figs.[17] and [18] are the results for CASE 1 and those for CASE 2 are shown in Fig.[19] and [20]. In the results of CASE 3, the 22 Hz component aliases as a 10 Hz component.







Figure 17 - Magnitude Plot Via FFT Algorithm, Case 1.



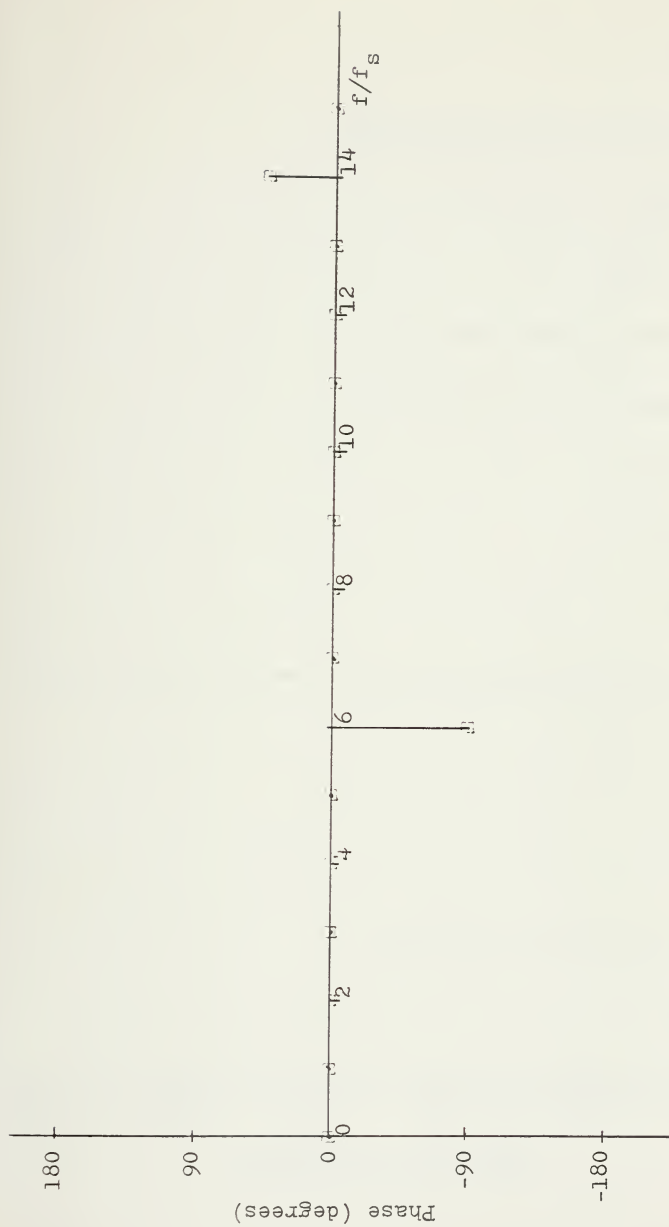


Figure 18 - Phase Plot Via FFT Algorithm, Case 1.



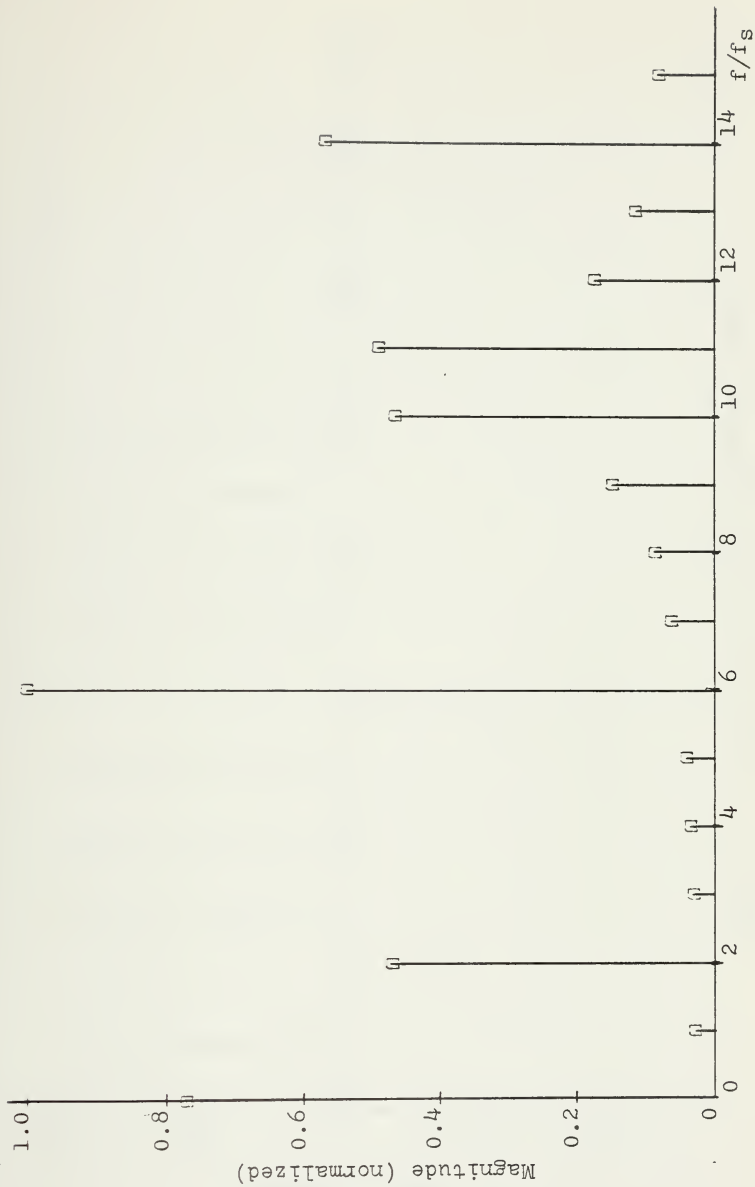


Figure 19 - Magnitude Plot Via FFT Algorithm, Case 2.



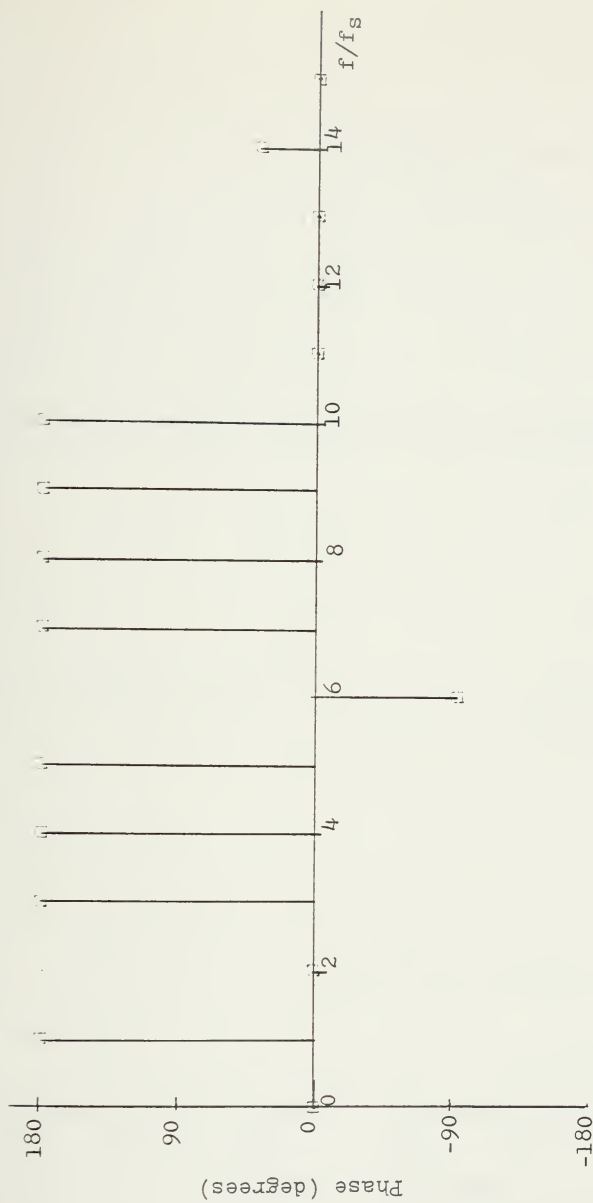


Figure 20 - Phase Plot Via FFT Algorithm, Case 2.





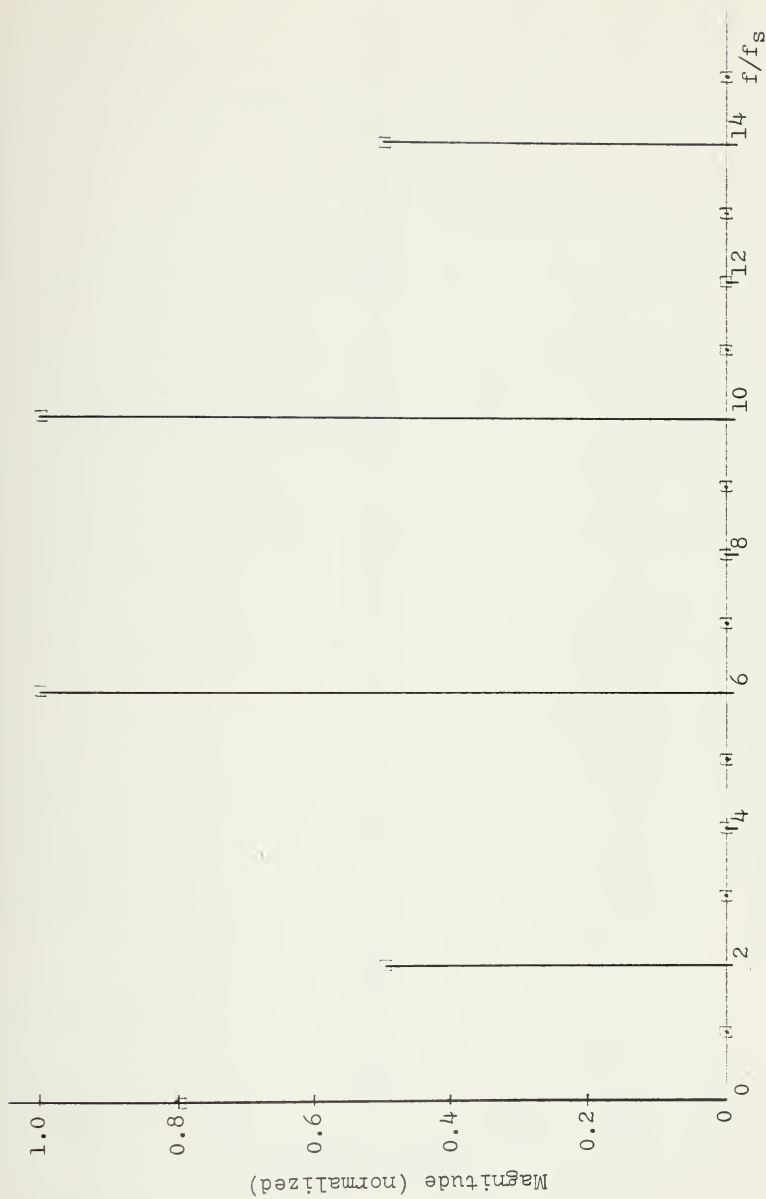


Figure 21 - Magnitude Plot Via FFT Algorithm, Case 3.



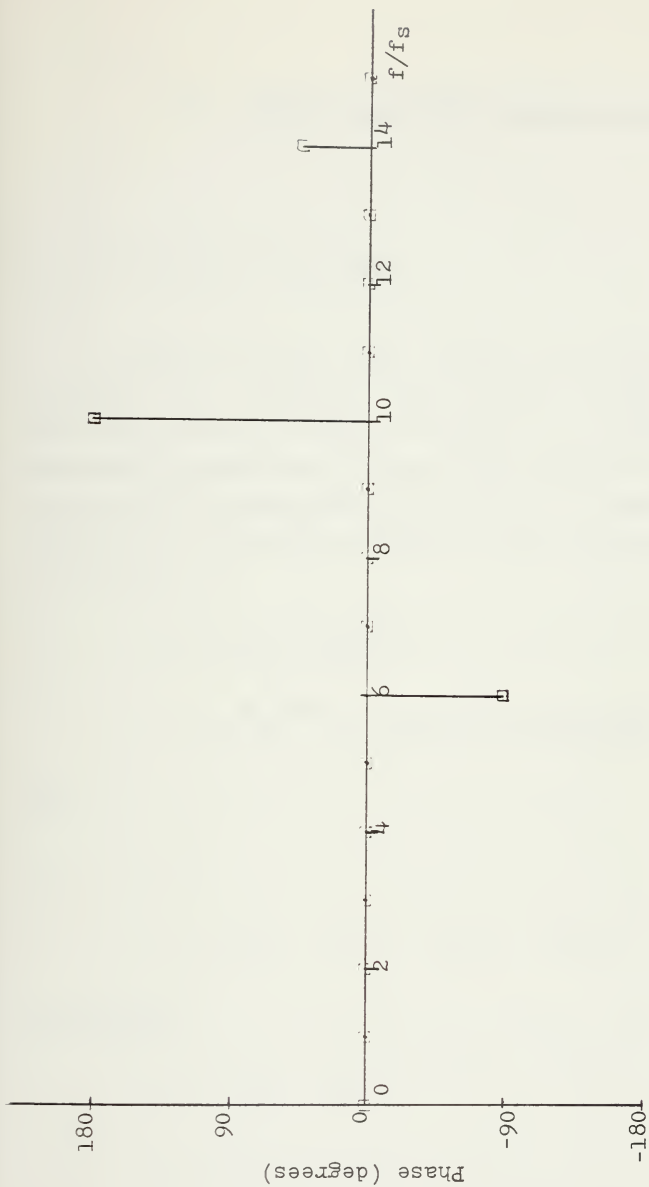


Figure 22 - Phase Plot Via FFT Algorithm, Case 3.



### III. CHIRP-Z-TRANSFORM COMPUTER SIMULATION

#### A. CZT PROGRAM DEVELOPEMENT

A computer program was written to simulate the CZT algorithm for computing the DFT of an arbitrary waveform. This waveform may be real or complex. In carrying out the processing operations, the real and imaginary components are represented by subscripts "r" and "i" respectively. The simulation program is a result of the following derivation. Proceeding, let

$x \rightarrow$  input sequence

$$f = A W^{-n} n^2/2 \rightarrow \text{complex pre-multiplier} \quad (3-1)$$

then

$$y = xf \quad (3-2)$$

$$y = x_r f_r + j x_r f_i + j x_i f_r + j^2 x_i f_i \quad (3-3)$$

simplifying

$$y_r = x_r f_r - x_i f_i \quad (3-4)$$

$$y_i = j(x_r f_i + x_i f_r) \quad (3-5)$$



Now letting

$$v = w^{-n^2/2} \quad (3-6)$$

$$g = y * v \quad (3-7)$$

where "\*" denotes convolution

$$g = y_r * v_r + y_r * jv_i + jy_i * v_r + jy_i * jv_i \quad (3-8)$$

$$g_r = y_r * v_r - y_i * v_i \quad (3-9)$$

$$g_i = y_r * jv_i + jy_i * v_r \quad (3-10)$$

In the program this convolution is simulated as a transversal filter with  $2N-1$  complex taps  $v_{-(N-1)}$  to  $v_{(N-1)}$  where

$$v_n = w^{-n^2/2}, \quad n = -(N-1) \text{ to } (N-1) \quad (3-11)$$

and

$$w = \exp(-j2\pi/r)$$

The third major step in the process, that of complex post-multiplication, is performed by letting

$$d = w^{k^2/2} \quad (3-12)$$





$$X = g d \quad (3-13)$$

$$X = g_r d_r + j g_r d_i + j g_i d_r + j^2 g_i d_i \quad (3-14)$$

simplifying

$$X_r = g_r d_r - g_i d_i \quad (3-15)$$

$$X_i = j(g_r d_i + g_i d_r) \quad (3-16)$$

For the special case of the DFT,  $A_o = 1$  and  $\theta_c = 0$  in Eq.[1-23],  $w_c = 1$  and  $\phi_c = -(1/N)$  in Eq.[1-24]. Thus  $A^{-n} = 1$ ,  $w = \exp(-j2\pi/N)$  and Eq.[3-1], [3-6], and [3-12] can be thought of as complex exponential sequences of linearly increasing frequency. They are represented as follows:

$$f_r = \cos(\pi n^2/N) \quad (3-17)$$

$$f_i = -\sin(\pi n^2/N) \quad (3-18)$$

$$v_r = \cos(\pi n^2/N) \quad (3-19)$$

$$v_i = \sin(\pi n^2/N) \quad (3-20)$$

$$d_r = \cos(\pi k^2/M) \quad (3-21)$$

$$d_i = -\sin(\pi k^2/M) \quad (3-22)$$



Substituting Eq.[3-17] thru [3-22] into Eqs.[3-4], [3-5], [3-9], [3-10], [3-15], and [3-16], respectively, describes the implementation of the CZT as shown pictorially in Fig.[23].

$$y_r = x_r \cos(\pi n^2/N) + x_i \sin(\pi n^2/N) \quad (3-23)$$

$$y_i = j[-x_r \sin(\pi n^2/N) + x_i \cos(\pi n^2/N)] \quad (3-24)$$

$$g_r = [y_r * \cos(\pi n^2/N)] - [y_i * \sin(\pi n^2/N)] \quad (3-25)$$

$$g_i = [y_i * j \sin(\pi n^2/N)] + [j y_r * \cos(\pi n^2/N)] \quad (3-26)$$

$$x_r = g_r \cos(\pi k^2/M) + g_i \sin(\pi k^2/M) \quad (3-27)$$

$$x_i = j[-g_r \sin(\pi k^2/M) + g_i \cos(\pi k^2/M)] \quad (3-28)$$

The simulation program is listed in Appendix D and is a working model of Fig.[23].



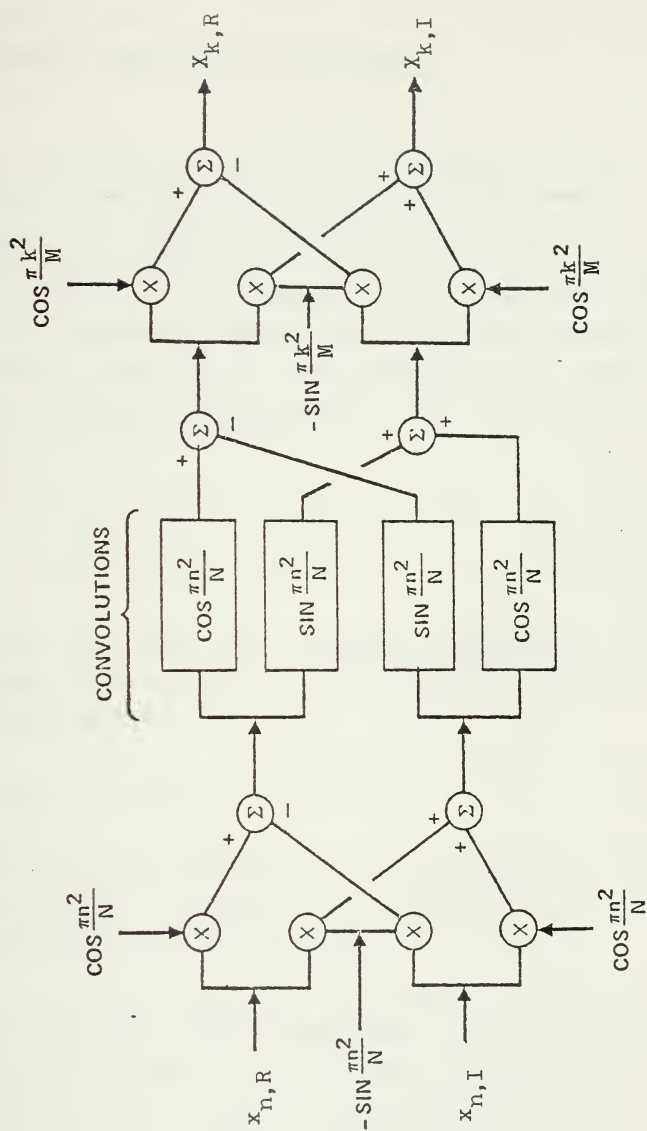


Figure 23 - DFT via CZT Algorithm with Parallel Implementation of Complex Arithmetic.



## B. EXAMPLES USING THE CZT ALGORITHM

As a test of the CZT algorithm to compute the DFT, CASES 1, 2, and 3 as described in Appendix A, were used as inputs for  $N=31$ . The imaginary component of the input is set equal to zero since the input consists of only real data. All figures are drawn with normalized frequency, but for the sake of discussion, assume that the input sinusoids have frequency in Hertz and the data window "T" is equal to 1 second. The data window used for all cases is rectangular. Thus the fundamental frequency is

$$F_o = \frac{1}{T} = \frac{1}{N(T_s)} = \frac{1}{(31)(1/31)} = 1 \text{ Hz}$$

which is in effect the minimum recognizable frequency component or resolution. The analyzing bandwidth or frequency range is

$$F_{\max} = (N/2)(F_o) = 15.5 \text{ Hz}$$

However, since the resolution is 1 Hz, the output components consist of the D-C component, the fundamental, and the first 14 harmonics for a frequency range of 15 Hz.

As in the FFT algorithm, the output of the CZT algorithm is a one-sided spectra consisting of  $[(N/2)+1]$  integer frequency components. Figs.[24] and [25] are the expected results for CASE 1 and Figs.[26] and [27] are the results for CASE 2. In the results of CASE 3, it should be noted that the aliasing frequency of 22 Hz folds back as a 9





Hz, whereas in the FFT it folded back as a 10 Hz component. This occurs because N is odd in the CZT where as in the FFT, N is required to be even and highly composite. For the FFT algorithm N = 32.



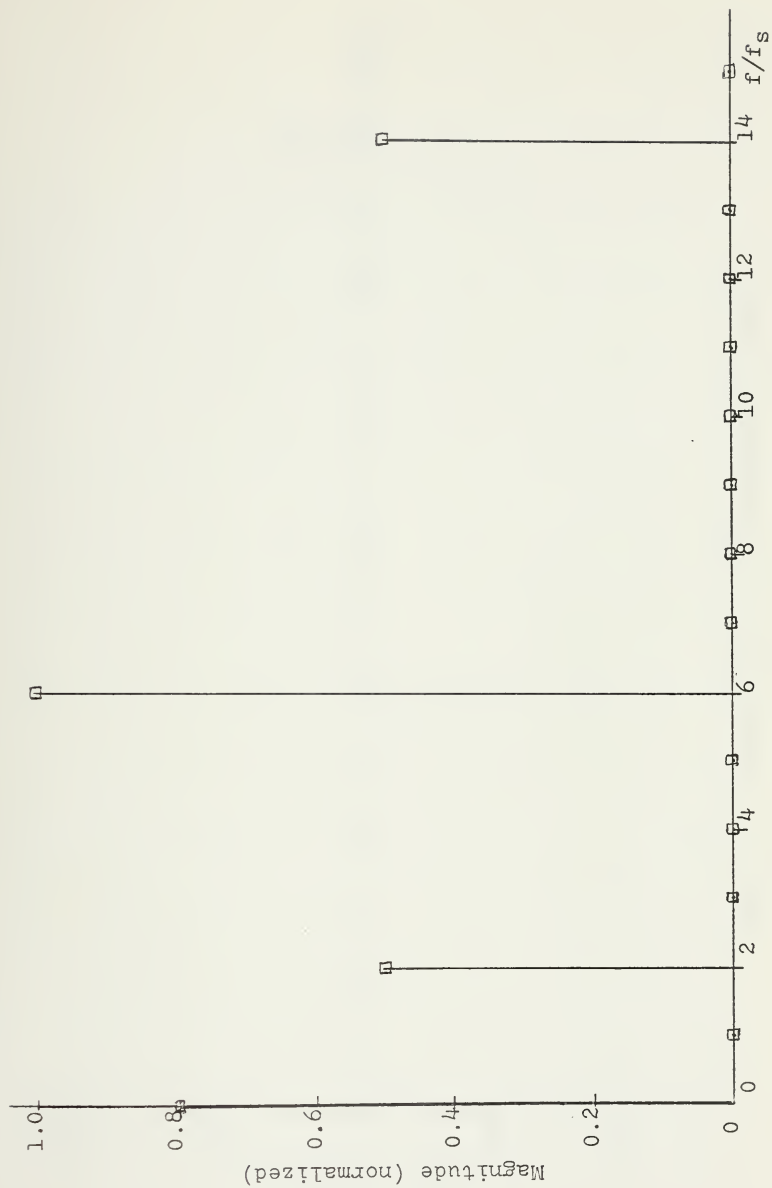


Figure 24 - Magnitude Plot Via CZT Algorithm, Case 1.



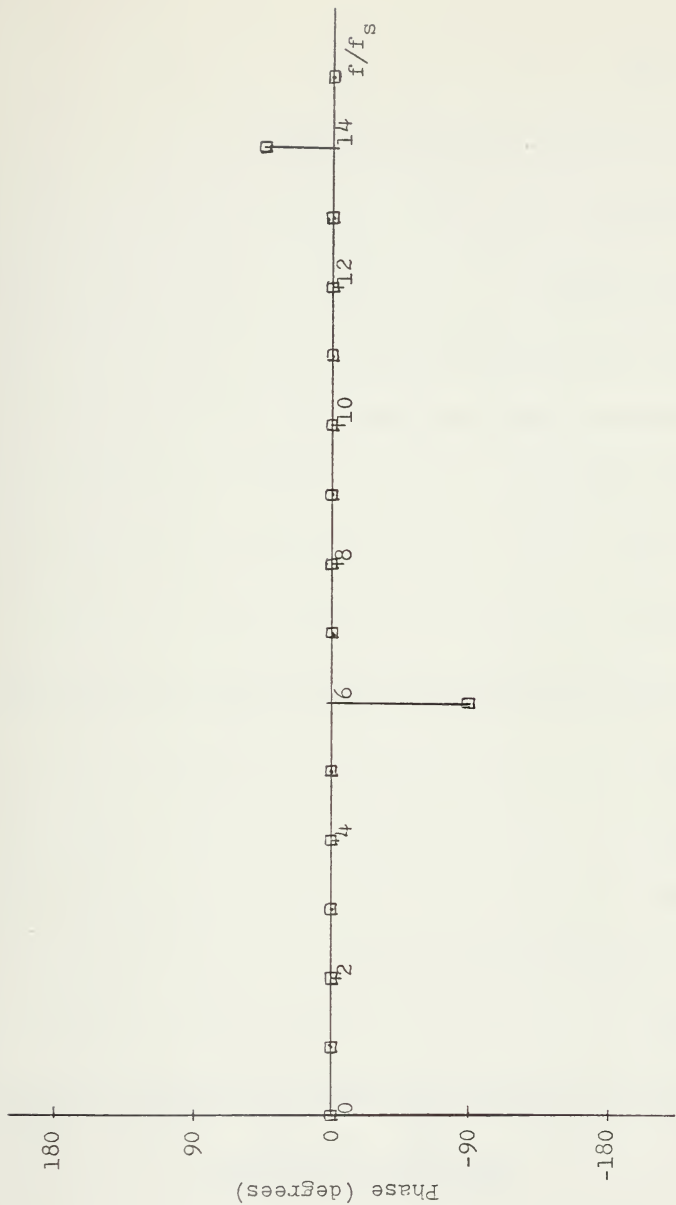


Figure 25 - Phase Plot Via CZT Algorithm, Case 1.



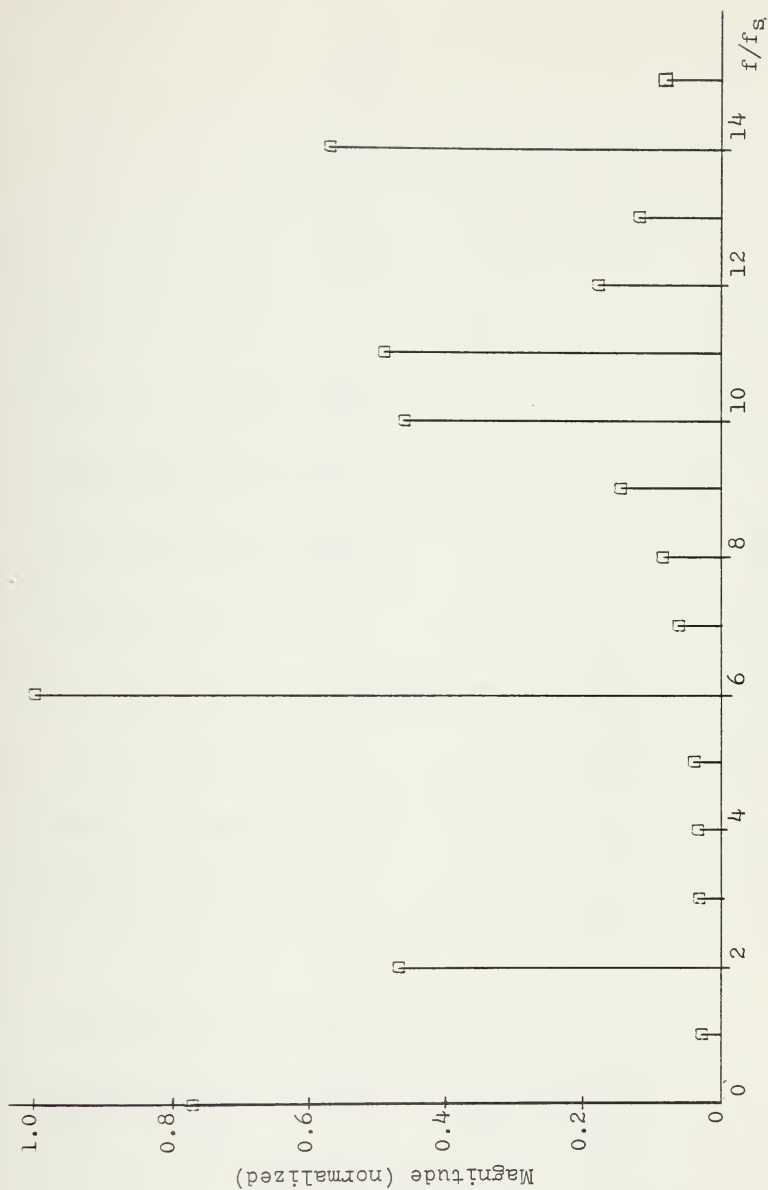


Figure 26 - Magnitude Plot Via CZT Algorithm, Case 2.





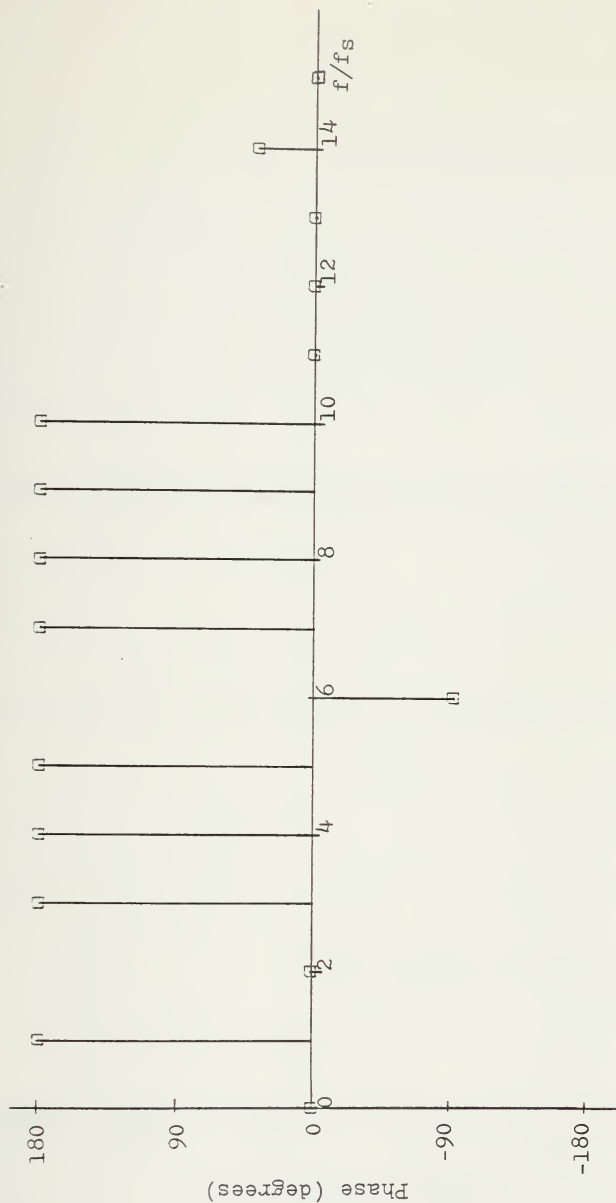


Figure 27 - Phase Plot Via CZT Algorithm, Case 2.





Figure 28 - Magnitude Plot Via CZT Algorithm, Case 3.



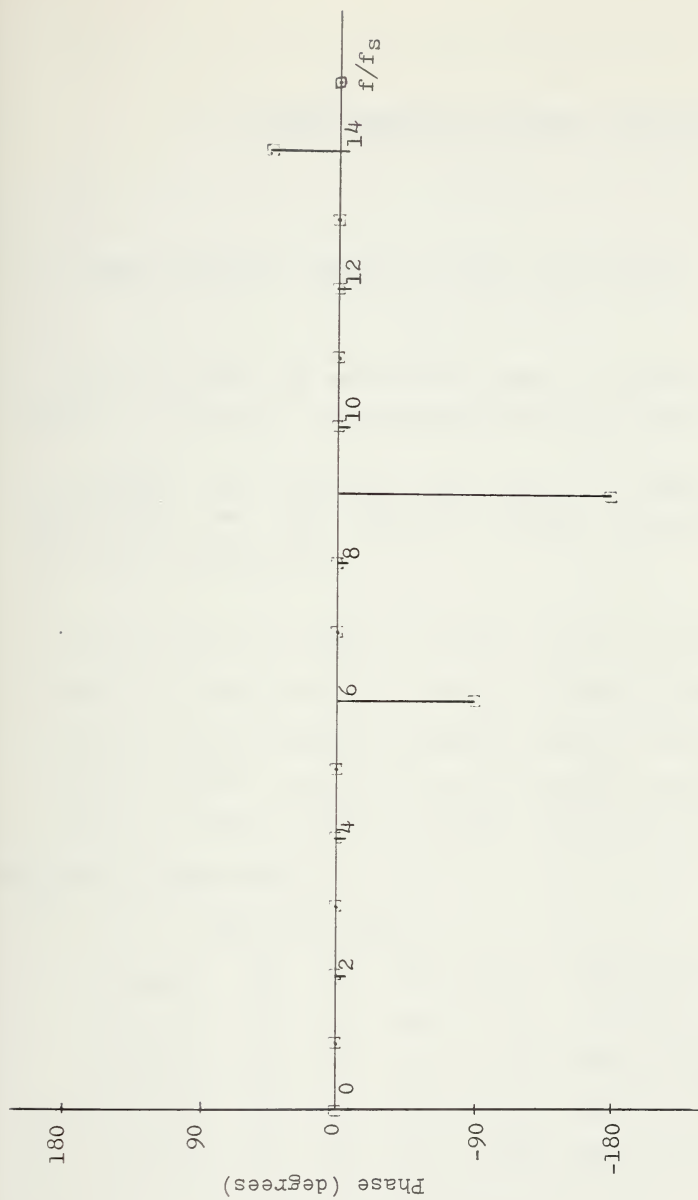


Figure 29 - Phase Plot Via CZT Algorithm, Case 3.



#### IV. PRIME TRANSFORM COMPUTER SIMULATION

##### A. PRIME TRANSFORM COMPUTER PROGRAM DEVELOPMENT

So as to make the mathematics clear, a simple example is given to illustrate the required matrices of the algorithm.

Consider the case  $N=5$  and let  $W = \exp(-j2\pi/N)$ . Evaluating the DFT equation directly results in the following  $N$  equations.

$$\begin{aligned} X(0) &= x(0)W^0 + x(1)W^0 + x(2)W^0 + x(3)W^0 + x(4)W^0 \\ X(1) &= x(0)W^0 + x(1)W^1 + x(2)W^2 + x(3)W^3 + x(4)W^4 \\ X(2) &= x(0)W^0 + x(1)W^2 + x(2)W^4 + x(3)W^6 + x(4)W^8 \quad (4-1) \\ X(3) &= x(0)W^0 + x(1)W^3 + x(2)W^6 + x(3)W^9 + x(4)W^{12} \\ X(4) &= x(0)W^0 + x(1)W^4 + x(2)W^8 + x(3)W^{12} + x(4)W^{16} \end{aligned}$$

Written in matrix form

$$\begin{aligned} X(k) &= [F] [x(n)] \\ \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix} &= \begin{bmatrix} W^0 & W^0 & W^0 & W^0 & W^0 \\ W^0 & W^1 & W^2 & W^3 & W^4 \\ W^0 & W^2 & W^4 & W^6 & W^8 \\ W^0 & W^3 & W^6 & W^9 & W^{12} \\ W^0 & W^4 & W^8 & W^{12} & W^{16} \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \end{bmatrix} \quad (4-2) \end{aligned}$$





The matrix  $\{F\}$  is written using the relationships

$$W^{nk} = W^{nk \bmod N}$$

$$W^0 = 1$$

For example  $W^9 = W^4$  since for  $n=3, k=3$

$$\begin{aligned} W^{nk} &= W^9 = \exp\left[(-j\frac{2\pi}{5})(9)\right] = \exp(-j\frac{18\pi}{5}) \\ &= \exp(-j\frac{8\pi}{5}) = \exp\left[(-j\frac{2\pi}{5})(4)\right] = W^4 \end{aligned} \quad (4-3)$$

therefore

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & W^1 & W^2 & W^3 & W^4 \\ 1 & W^2 & W^4 & W^1 & W^3 \\ 1 & W^3 & W^1 & W^4 & W^2 \\ 1 & W^4 & W^3 & W^2 & W^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \end{bmatrix} \quad (4-4)$$

Then deleting  $X(0)$  and  $x(0)$ , Eq. [4-4] is

$$\begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \end{bmatrix} - x(0) \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} W^1 & W^2 & W^3 & W^4 \\ W^2 & W^4 & W^1 & W^3 \\ W^3 & W^1 & W^4 & W^2 \\ W^4 & W^3 & W^2 & W^1 \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \\ x(3) \\ x(4) \end{bmatrix} \quad (4-5)$$

$X'$                        $D$                        $F'$                        $x'(n)$

For  $N=5$  and  $r=2$ , there is a one to one mapping of the integers  $n, k$  to the integers  $(n)_p, (k)_p$ , i.e.,

$$2^1 = 2 \bmod 5, \quad 2^2 = 4 \bmod 5, \quad 2^3 = 3 \bmod 5, \quad 2^4 = 1 \bmod 5$$

$n, k$	1	2	3	4
$(n)_p, (k)_p$	2	4	3	1

(4-6)



Matrix  $\{F'\}$  is factored into three matrices.

$$F' = P^t C P$$

The elements of the  $\{C\}$  matrix are found by using the relationship given by Eq. [1-42]. Thus

$$C = \begin{bmatrix} w^4 & w^3 & w^1 & w^2 \\ w^3 & w^1 & w^2 & w^4 \\ w^1 & w^2 & w^4 & w^3 \\ w^2 & w^4 & w^3 & w^1 \end{bmatrix} \quad (4-7)$$

The elements of the permutation matrix are given by Eq. [1-43].

$$P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad (4-8)$$

The elements of the transpose of  $P$  are

$$P^t = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad (4-9)$$

A functional diagram of the prime transform algorithm for computing the Fourier coefficients of real data is as shown in Fig. [30]. The switch is in the up position for the first data sample and is down for the remaining  $(N-1)$  samples.



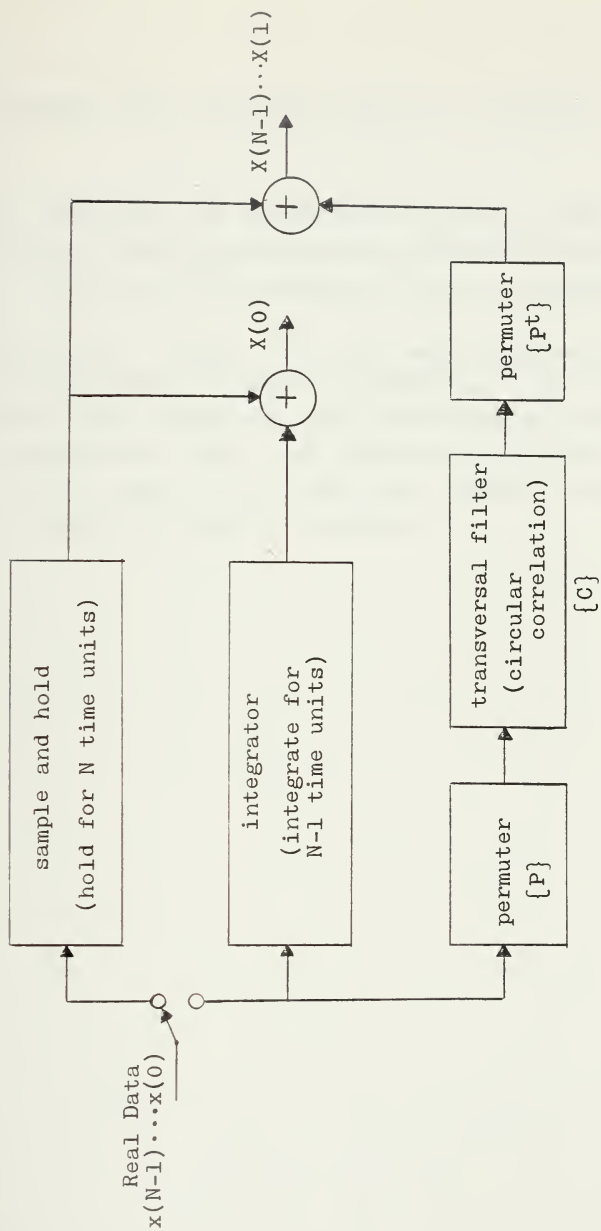


Figure 30 - Functional Diagram of the Prime Transform Algorithm.



## B. EXAMPLES USING THE PRIME TRANSFORM ALGORITHM

To simulate the operations of the prime transform algorithm, a computer program was written for the odd prime,  $N=31$ . A listing of the program is given in Appendix E.

Once again Cases 1, 2, and 3 were employed in order to evaluate the prime transform algorithm. The fundamental frequency and frequency range are the same as that of the CZT algorithm for  $N=31$ . The results are shown in Figs. [31] thru [36]. As can be seen, the results are the same as those computed by the CZT algorithm.





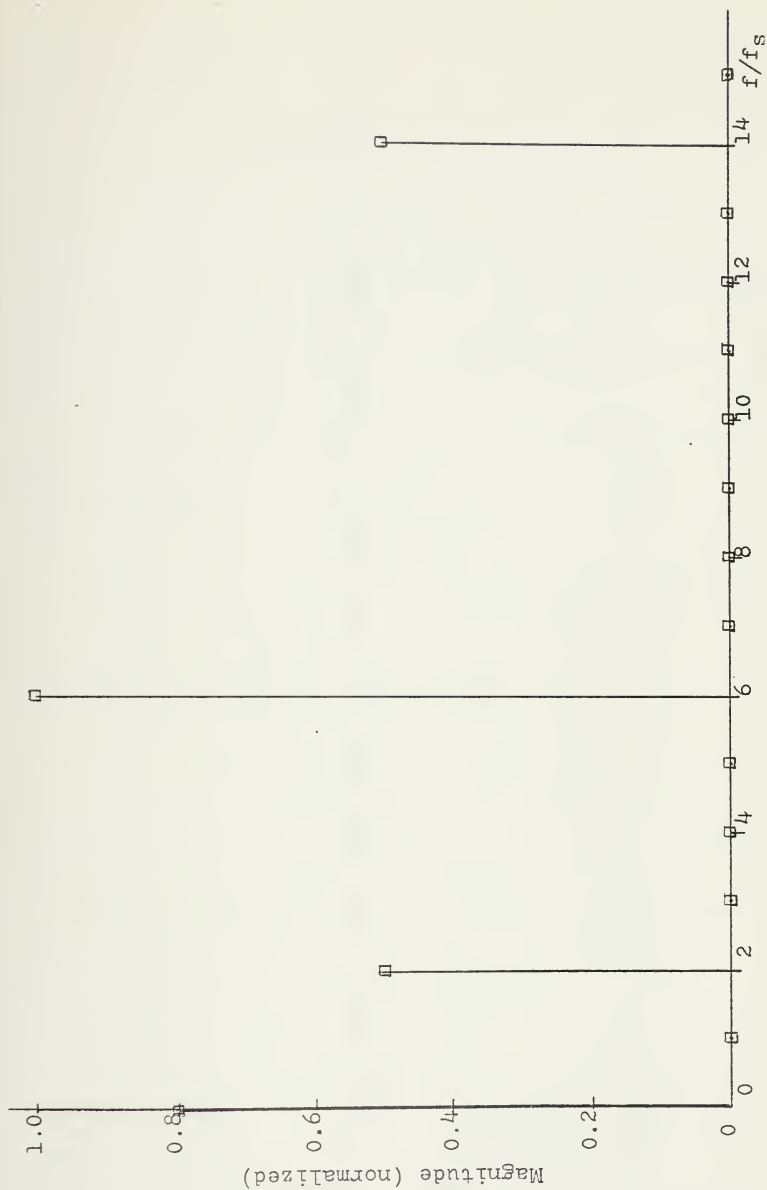


Figure 31 - Magnitude Plot Via Prime Transform Algorithm, Case 1.



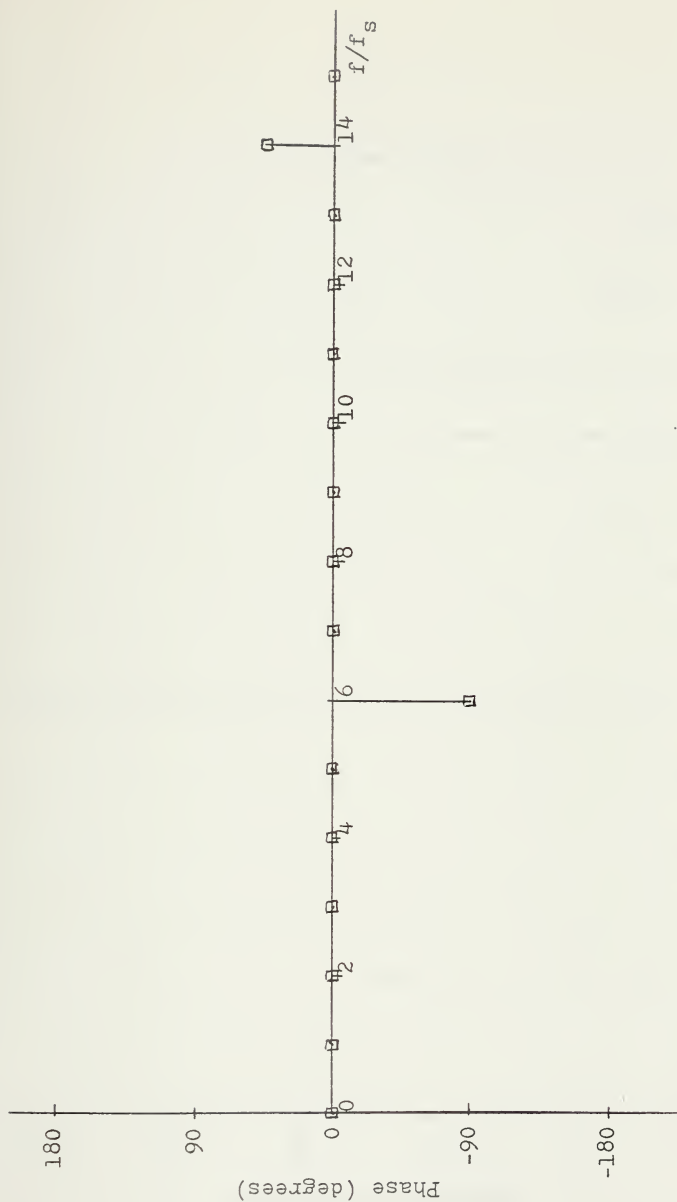


Figure 32 - Phase Plot Via Prime Transform Algorithm, Case 1.



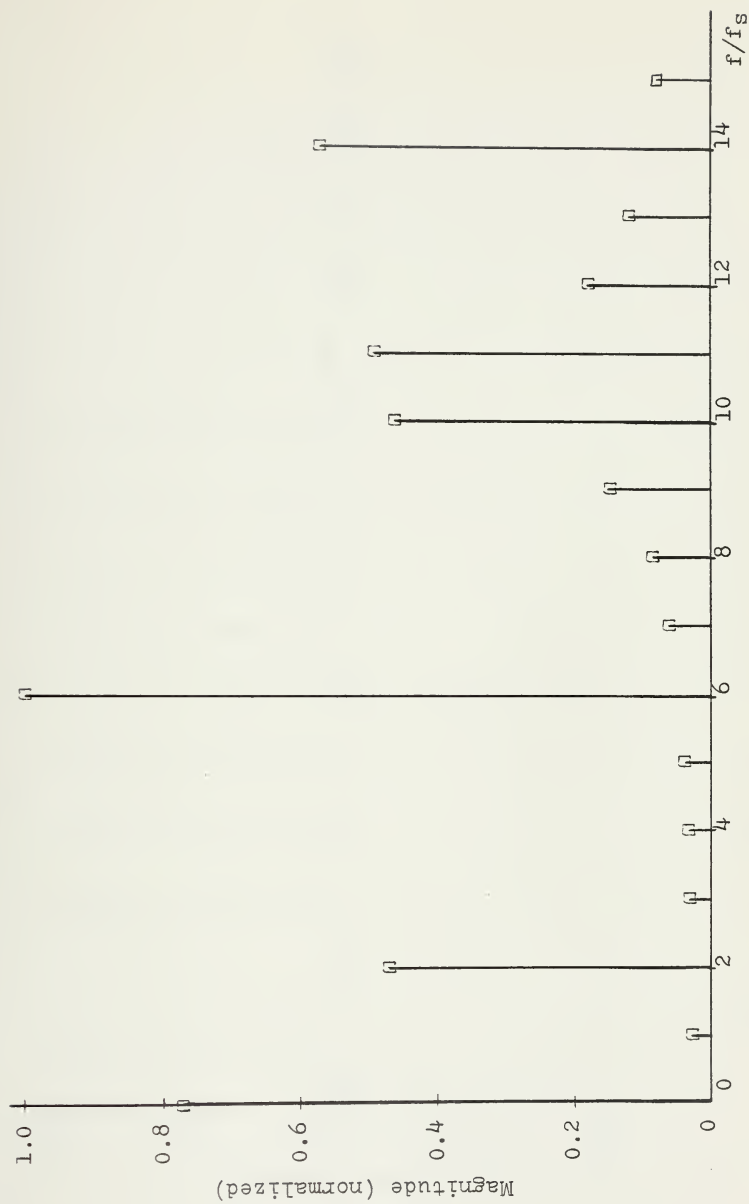


Figure 33 - Magnitude Plot Via Prime Transform Algorithm, Case 2.



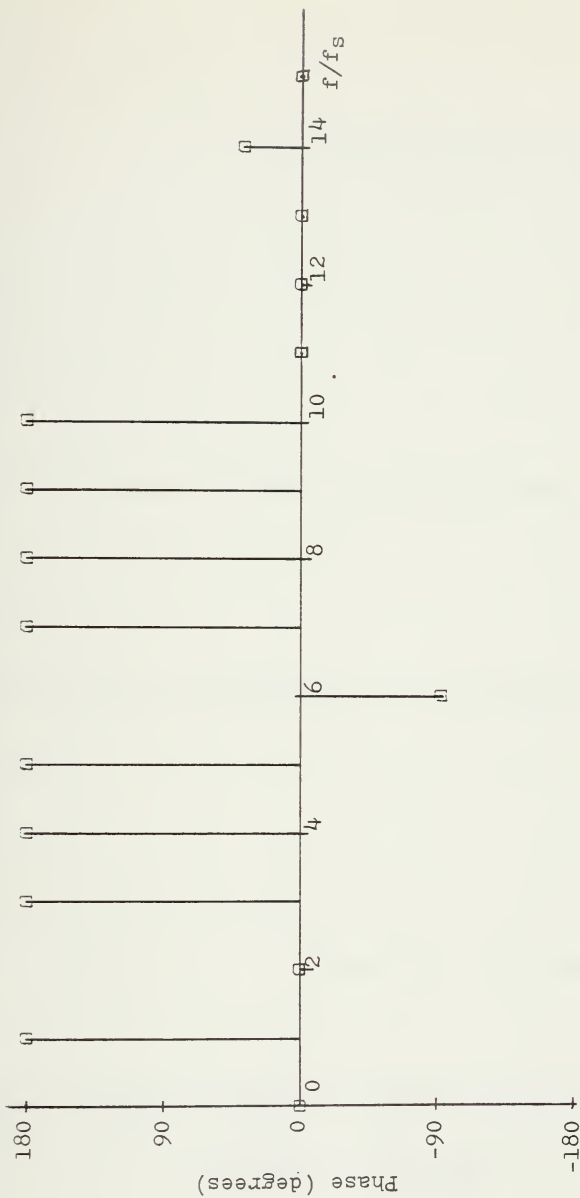


Figure 34 - Phase Plot Via Prime Transform Algorithm, Case 2.





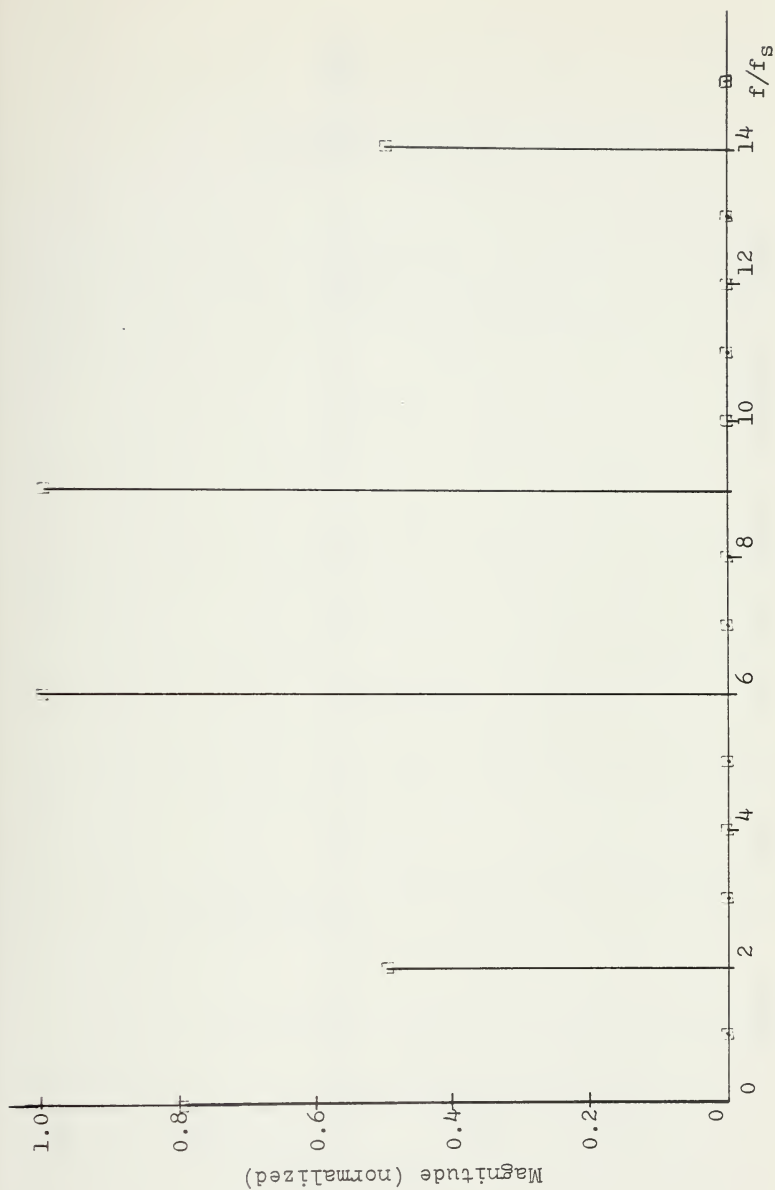


Figure 35 - Magnitude Plot Via Prime Transform Algorithm, Case 3.



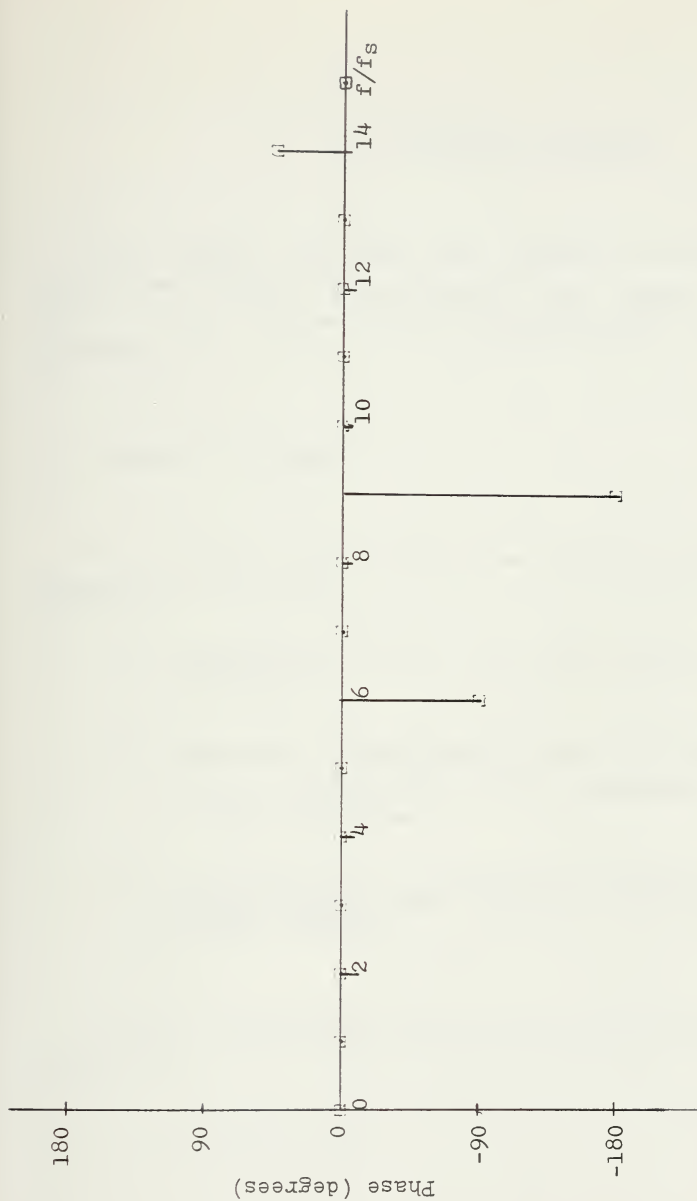


Figure 36 - Phase Plot Via Prime Transform Algorithm, Case 3.



## V. CZT SENSITIVITY ANALYSIS

This chapter addresses the sources of error in both digital and sampled analog systems with special attention given to the sampled analog implementation of the CZT algorithm.

### A. DIGITAL FFT SYSTEM

In a digital FFT system, the sources of error may be summarized as follows:

1. Quantization of the data at the input A/D converter.
2. Inaccurate transformation due to errors in finite representation of the coefficients  $W_N^k$  in the butterflies.
3. Roundoff noise in truncating the result of a multiplication and scaling the data to prevent overflow.

An rms quantization error in the A/D converter can be defined as

$$Q_e = 2^{-b/12} \quad (5-1)$$



where  $b_1$  is the number of bits used to represent the input data. This noise does not scale with signal size and dominates only at low signal levels.

The error due to the inexact representation of the multiplier coefficients  $\frac{k}{N}$  can be expressed as the ratio of mean square output error to mean square output signal to give

$$E_e = \left( \frac{\log_2 N}{6} \right)^2 2^{-(2)(b_3)} \quad (5-2)$$

where  $b_3$  is the number of bits used to represent the FFT coefficients. This equation predicts that the variance of the error increases at a rather slow rate with  $N$ .

A measure of the rms noise output to the rms signal output due to errors produced by roundoff and signal truncation to prevent overflows in the butterflies is given by

$$\frac{\text{rms}(\text{error})}{\text{rms}(\text{signal})} = \frac{\sqrt{N} 2^{-b_2} (0.3)^8}{\text{rms}(\text{input})} \quad (5-3)$$

where  $b_2$  is the number of bits used to represent the result of the operation. This upper bound of the error-to-signal is seen to increase as  $\sqrt{N}$ , and is the most important source of error in a digital FFT.





## E. SAMPLED ANALOG CZT SYSTEM

Instead of errors resulting from registers containing a finite word length, as in the digital FFT system, the output accuracy of the sampled analog CZT system is limited by the CCD's and SAW's sensitivity to the environment (temperature, humidity, etc.,) and fabrication imperfections. The sources of error of a CCD CZT system using pre- and post-multiplying coefficients stored in either a ROM or PROM are as follows

1. Thermal noise  $\rightarrow$  this source of error is analogous to quantization in a digital FFT because it generates an error which is independent of signal level.
2. The errors introduced by inaccuracies in the pre-and post-multipliers.
3. The errors introduced by the inaccuracies in the tap weights of the transversal filter.
4. The error due to mismatch and nonlinearities in the differential current amplifiers.
5. The error due to mismatches of the post-multiplier outputs.

The errors addressed in this study were those due to inaccuracies in the pre- and post-multipliers and the inaccuracies in the tap weights of the transversal filter. The errors in the pre- and post-multipliers were modeled as being a percentage "p" of the theoretical coefficient value  $m(n)$ , i.e.,

$$y = p[m(n)] \quad (5-4)$$



and to be uniformly distributed on  $(-\beta/2, \beta/2)$ . The transversal filter was modeled as a tapped analog delay line with the tap weights determined by the value of the external resistors. The error in the value of the tap weight was modeled as being a percentage "p" of the theoretical tap weight value  $h(n)$ , i.e.,

$$\beta = F[h(n)] \quad (5-5)$$

and to be uniformly distributed on  $(-\beta/2, \beta/2)$ . These error models were included in the simulation program [Appendix D].

### C. SENSITIVITY TESTS

Three different input waveforms were used to evaluate the performance of the CZT algorithm for the following percentages

$$p = 0\%, 4\%, 10\%, 20\%, 30\%, \text{ and } 40\%$$

In all three tests  $N=512$  and a rectangular data window equal to 4 seconds was used. The value of 4 seconds was used for ease of plotting the output spectrum. Thus the frequency resolution is  $f/f_s = 0.25$  and the frequency range is  $f/f_s = 64.0$ .

The first waveform consisted of the impulse function shown in Fig.[37]. The time impulse responses of the cosine and sine filters of the convolver for the different percentages are shown in Figs.[38] thru [50]. For  $p=0\%$ , Fig.[51] shows the expected results, i.e., equal magnitude frequency components covering the entire frequency range. As the percentage of error increases, examination of



the figures reveals that the average value of the magnitude of the frequency components is greater than 1.0.



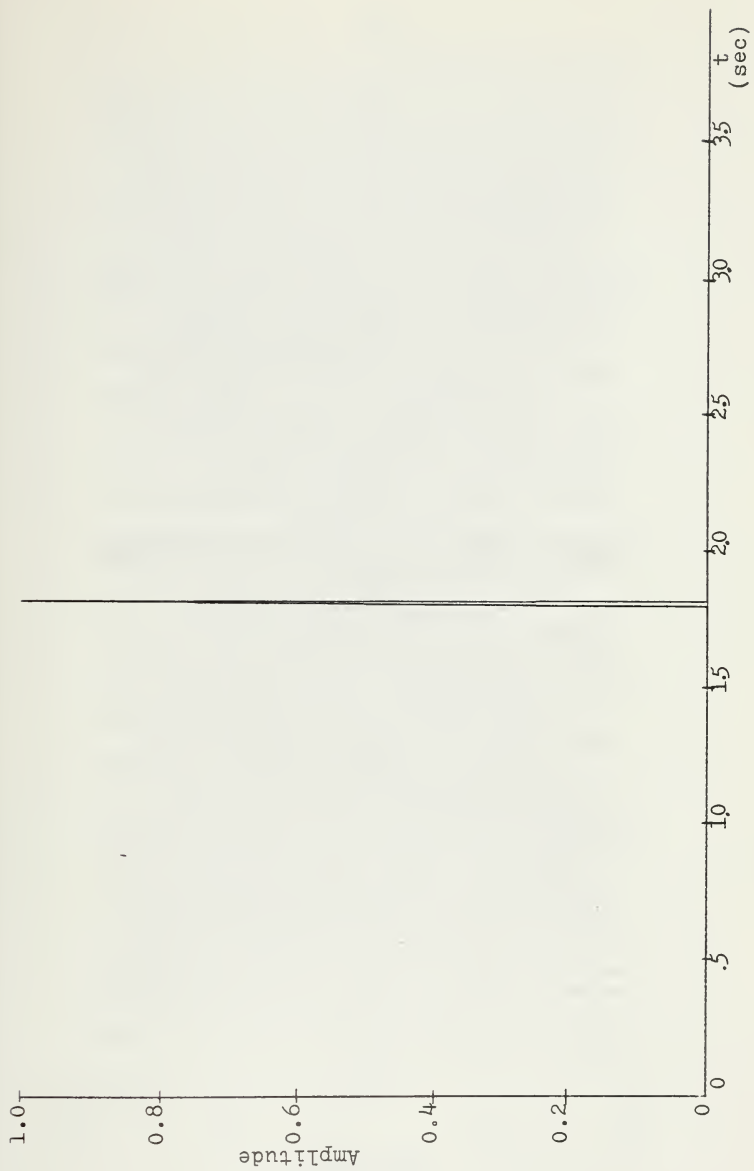


Figure 37 - Impulse Input (normalized  $A/N$ ).





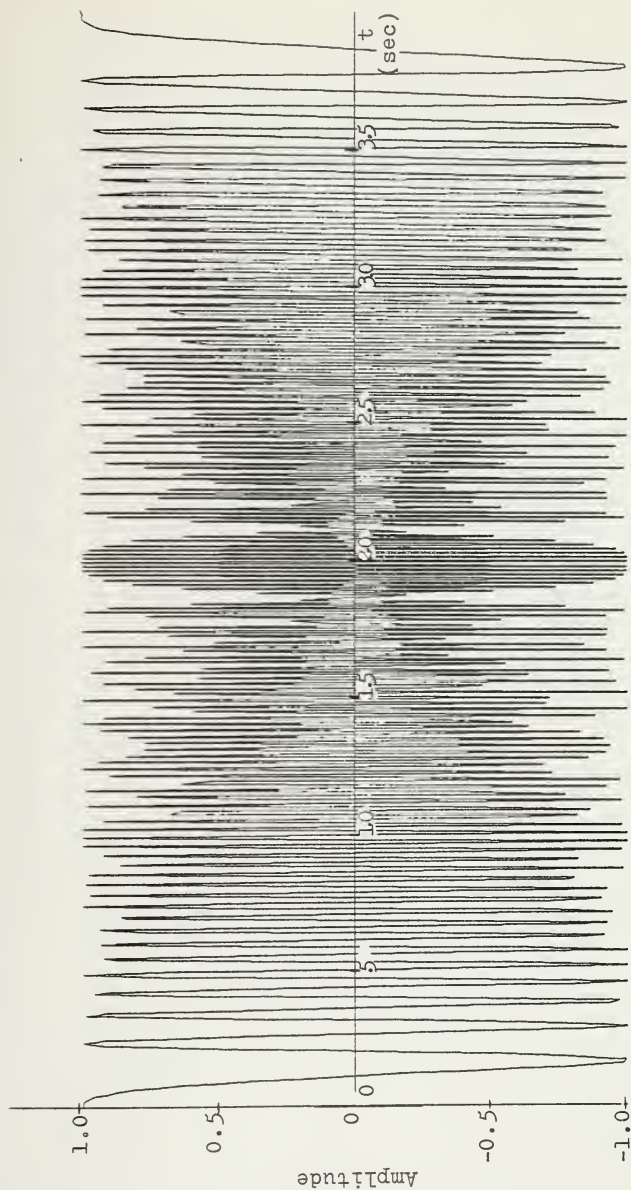


Figure 38 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=0\%$ .



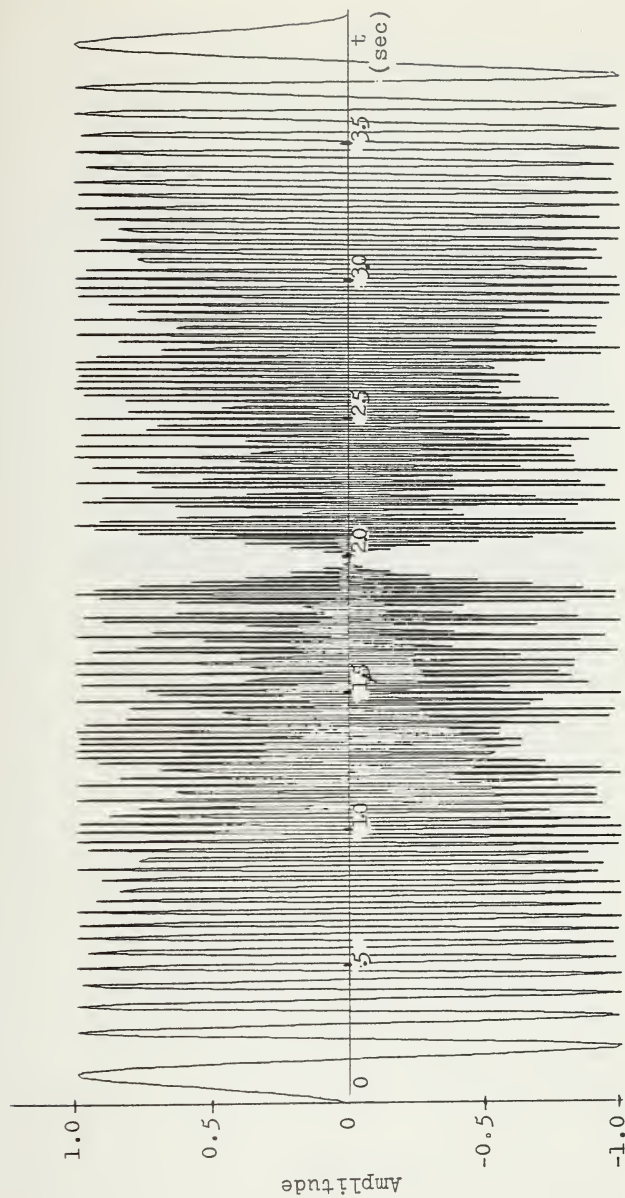


Figure 39 - Time Impulse Response of the Sine Filter,  $N=512$ ,  $p=0\%$ .



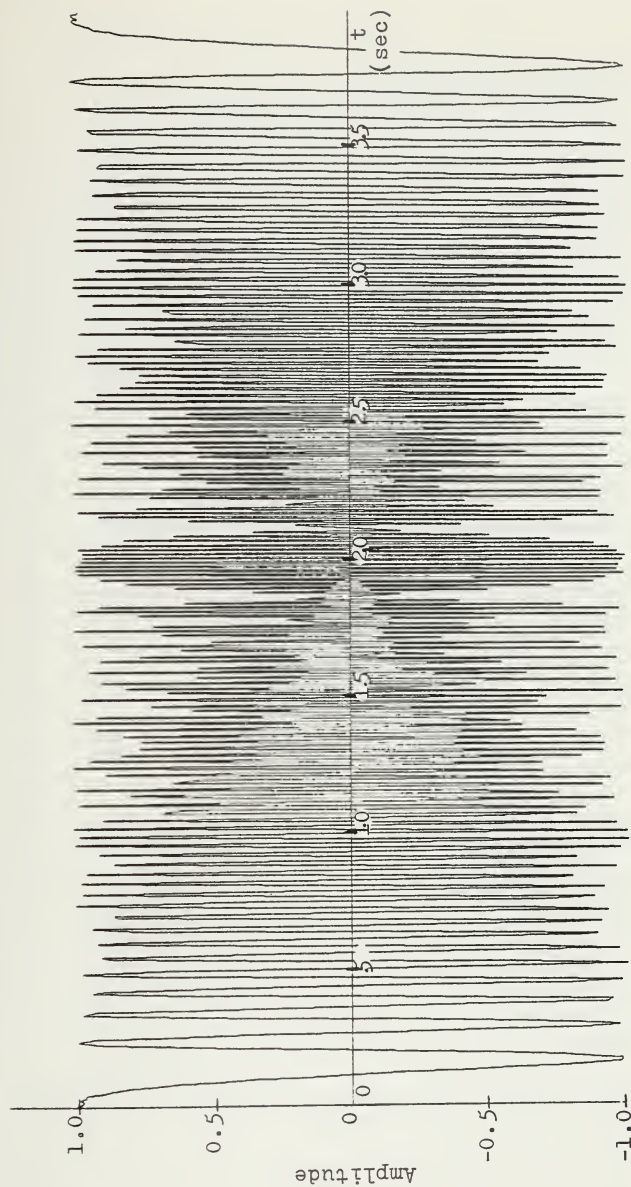


Figure 40 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=4\%$



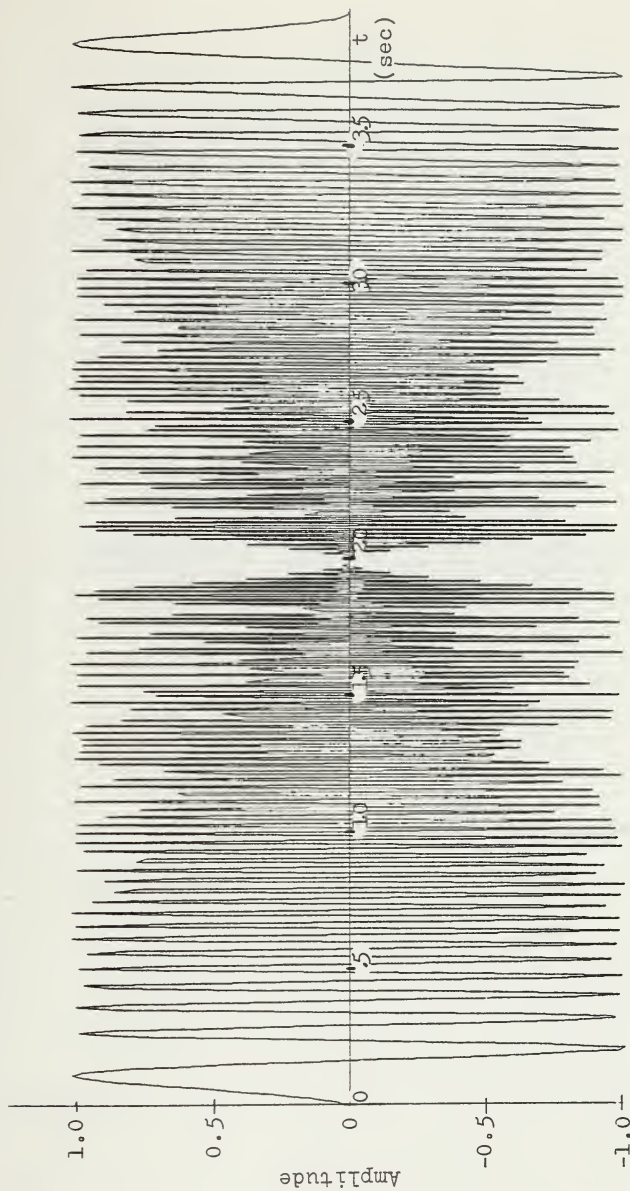


Figure 41 - Time Impulse Response of the Sine Filter;  $N=512$ ,  $p=4\%$ .





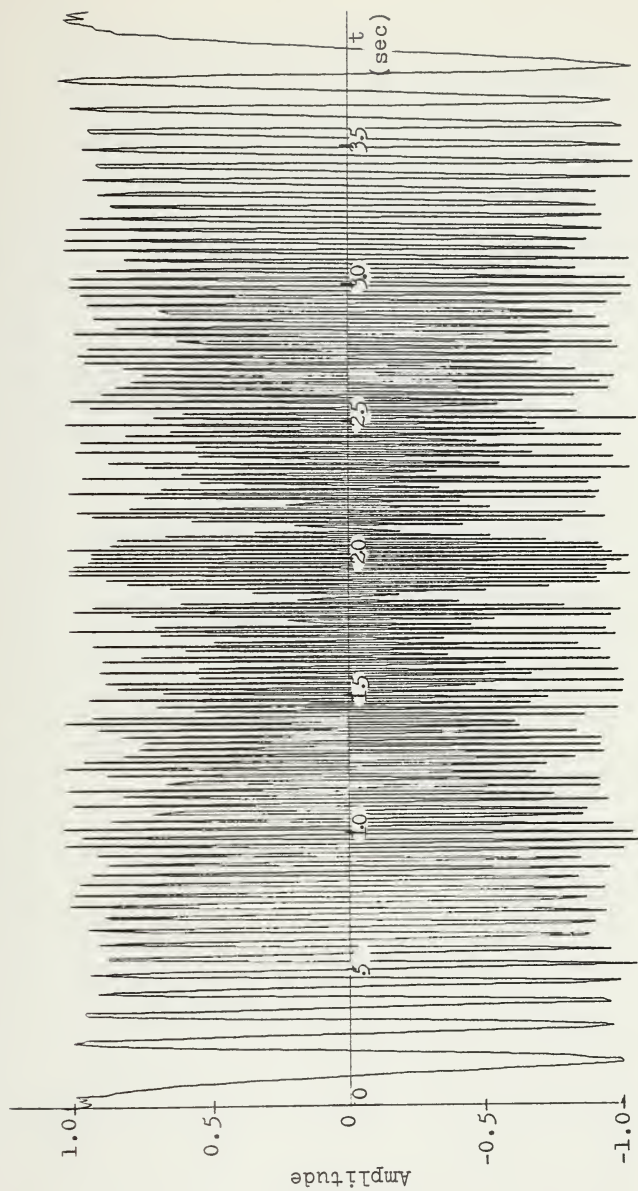


Figure 42 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=10\%$ .



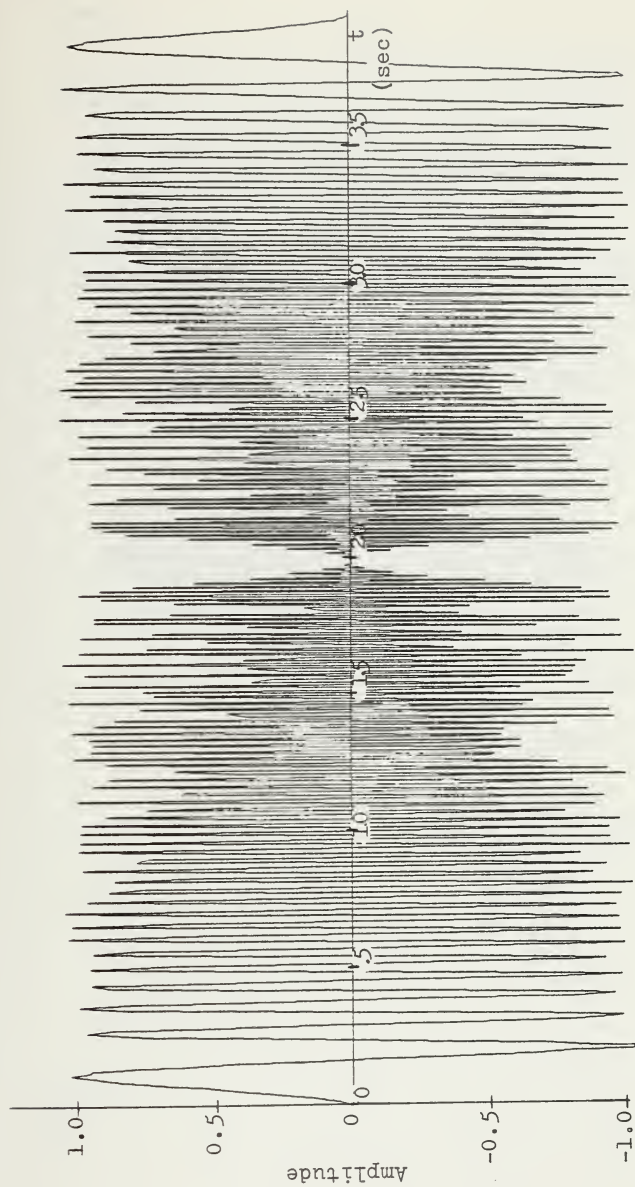


Figure 43 - Time Impulse Response of the Sine Filter;  $N=512$ ,  $p=10\%$ .



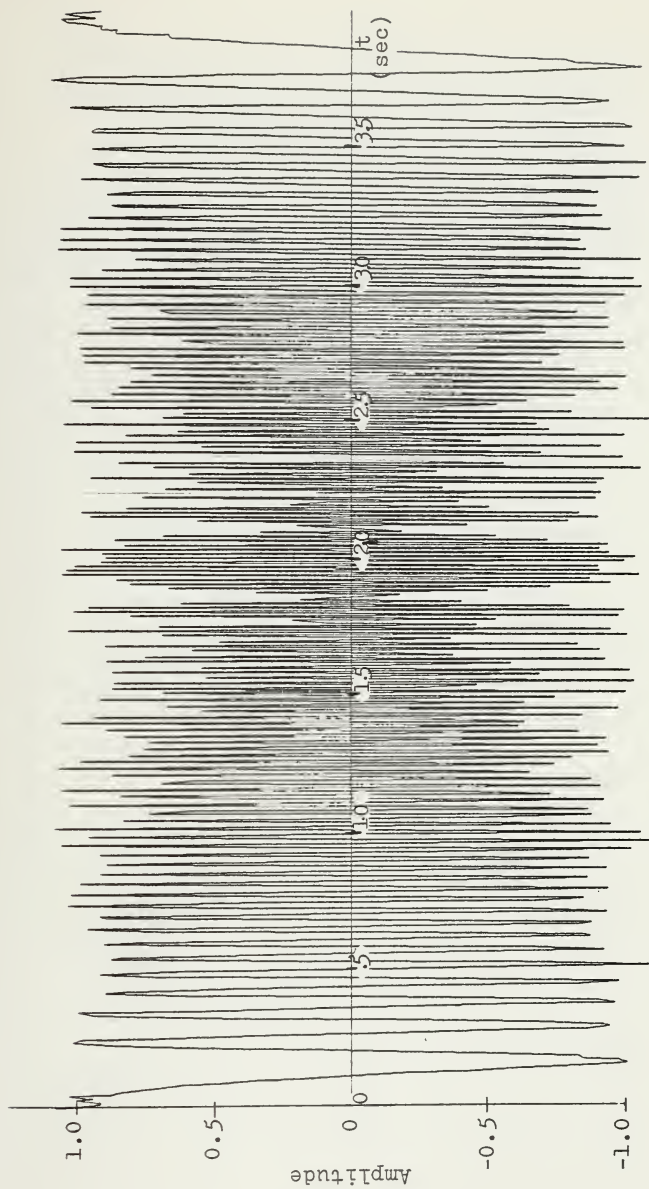


Figure 44 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=20\%$ .



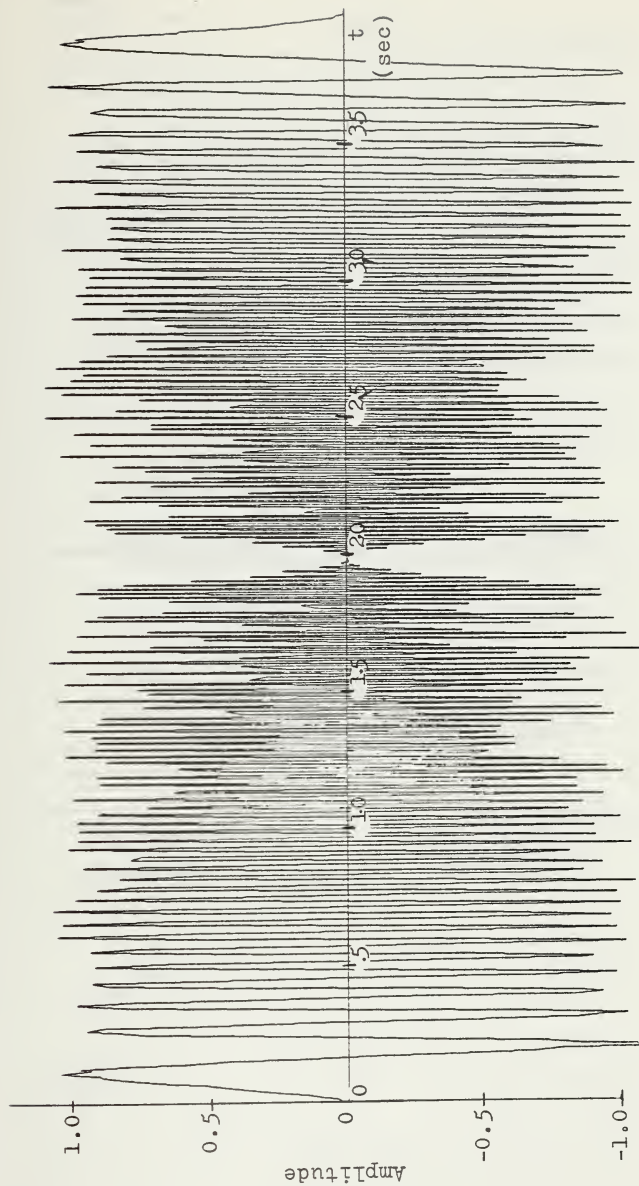


Figure 45 - Time Impulse Response of the Sine Filter;  $N=512$ ,  $p=20\%$ .





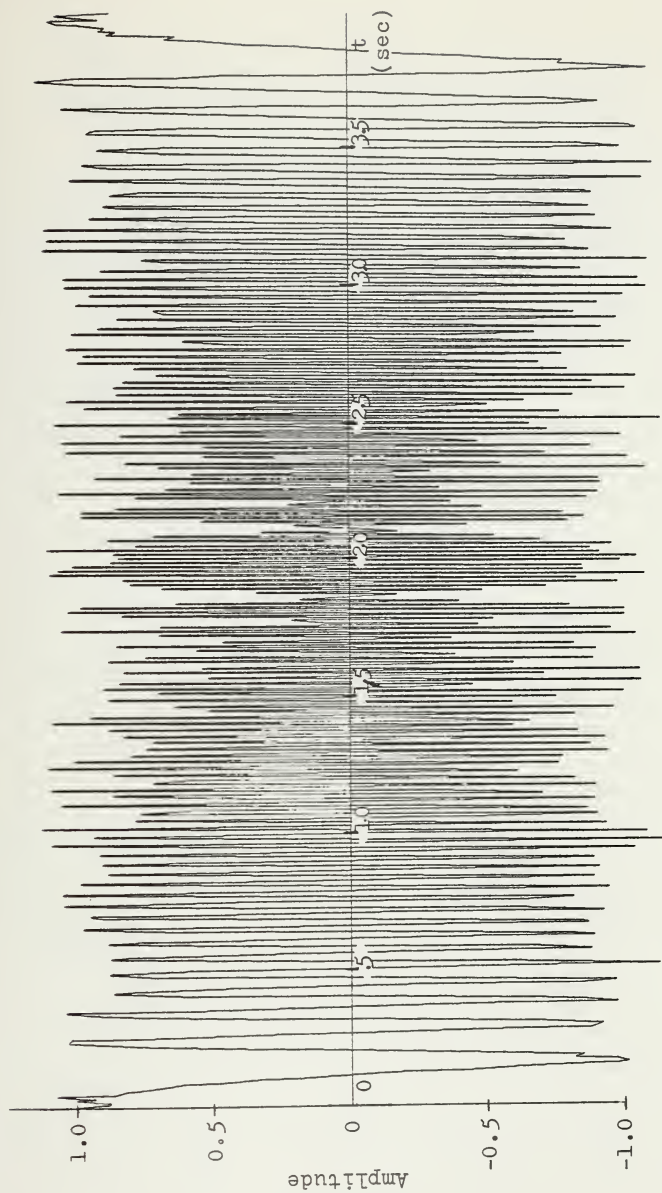


Figure 46 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=30\%$ .



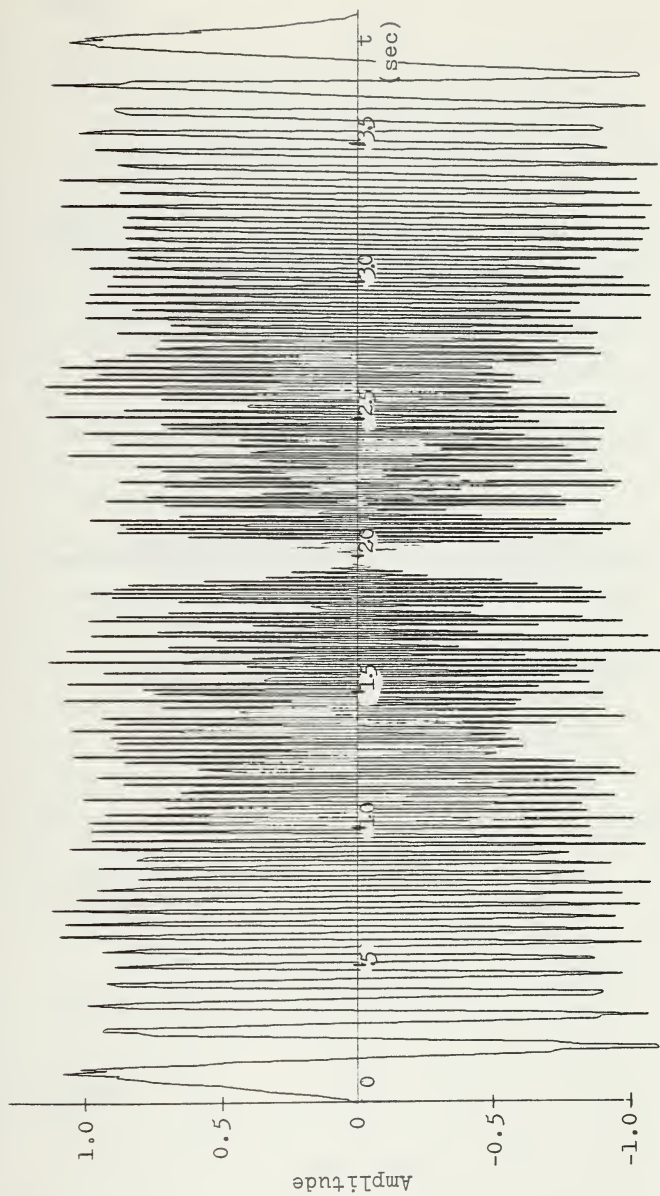


Figure 47 - Time Impulse Response of the Sine Filter;  $N=512$ ,  $p=30\%$ .



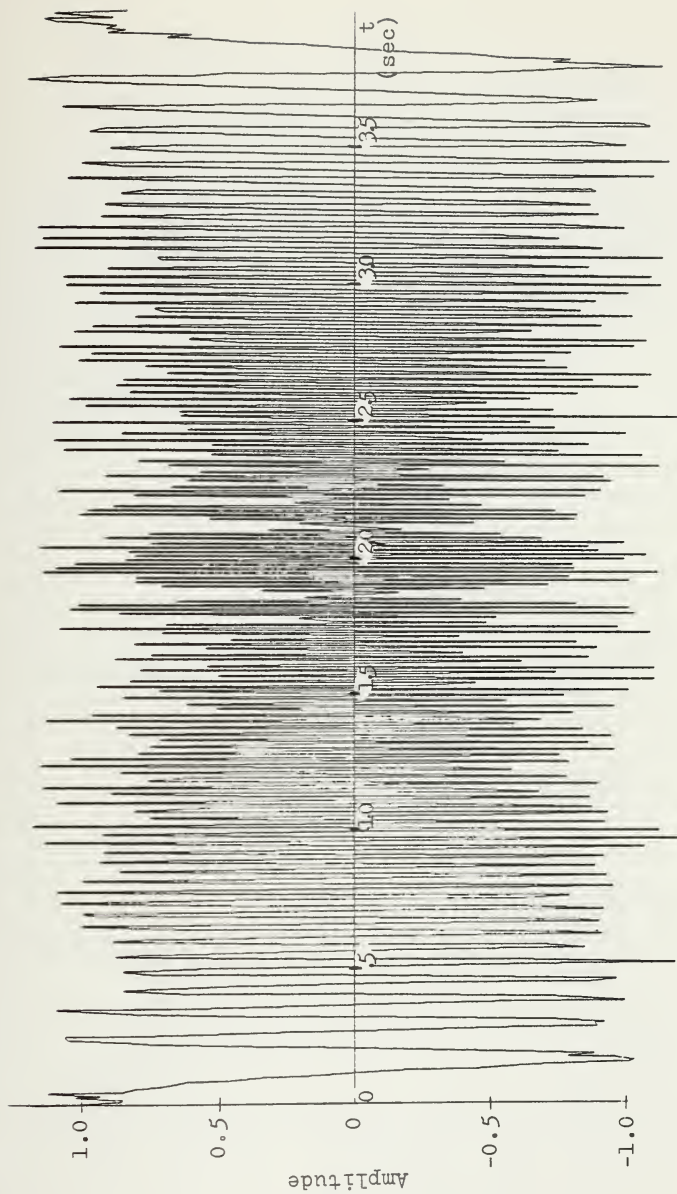


Figure 48 - Time Impulse Response of the Cosine Filter;  $N=512$ ,  $p=40\%$ .



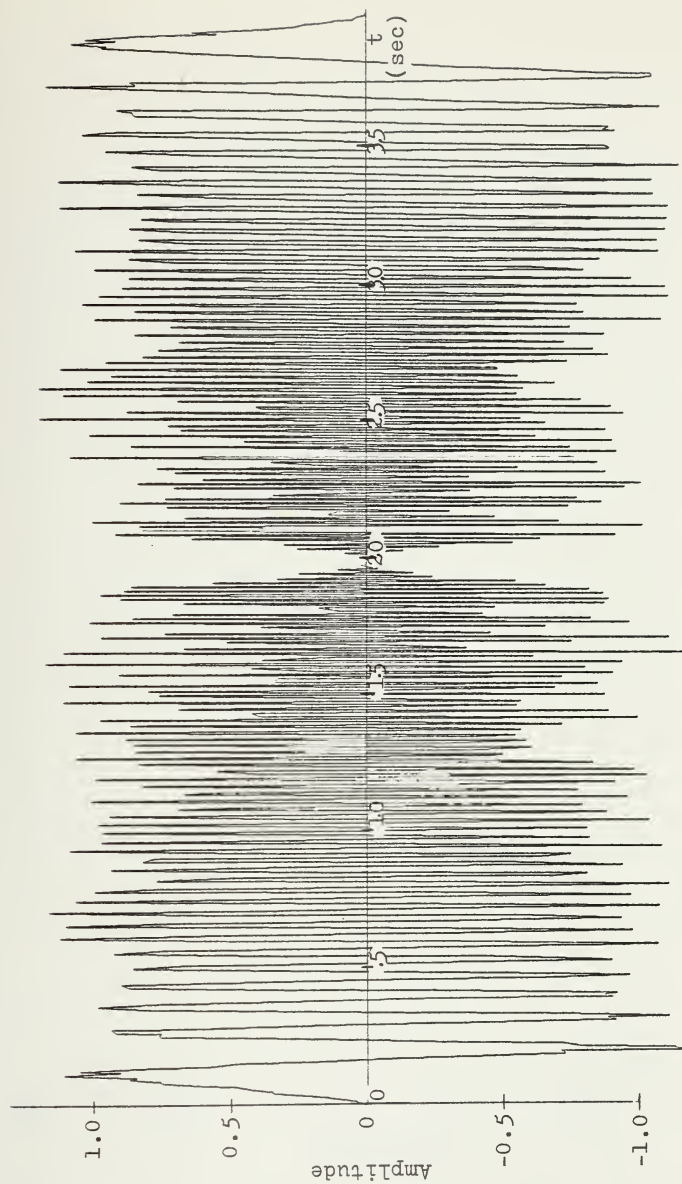


Figure 49 - Time Impulse Response of the Sine Filter;  $N=512$ ,  $p=40\%$ .







Figure 50 - Magnitude Plot of Input Impulse,  $p=0\%$ .





Figure 51 - Magnitude Plot of Input Impulse,  $p=4\%$ .



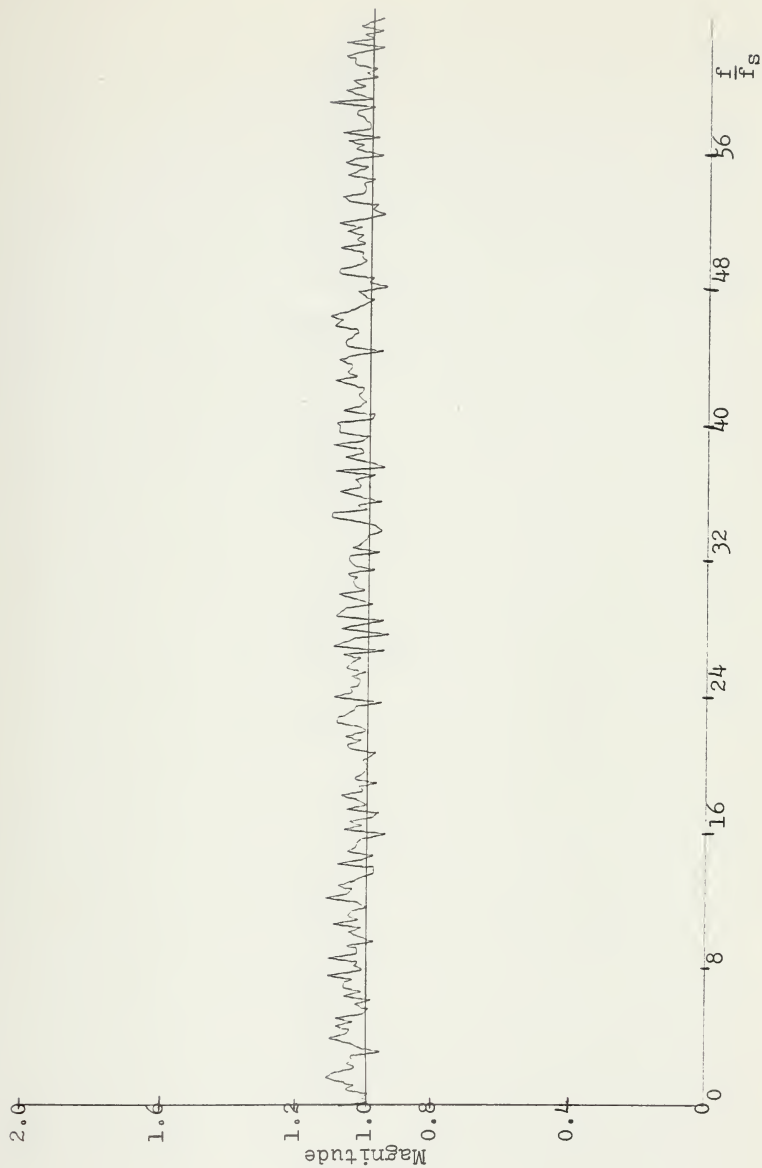


Figure 52 - Magnitude Plot of Input Impulse,  $p=10\%$ .



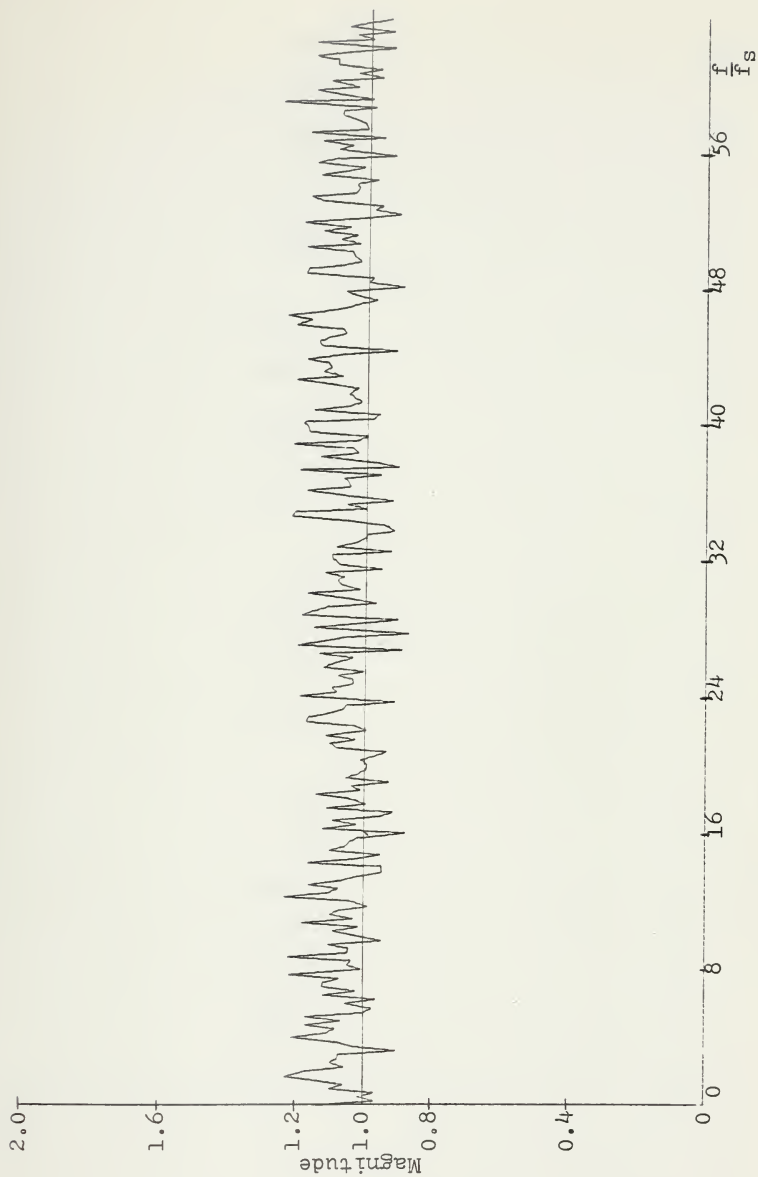


Figure 53 - Magnitude Plot of Input Impulse,  $p=20\%$ .





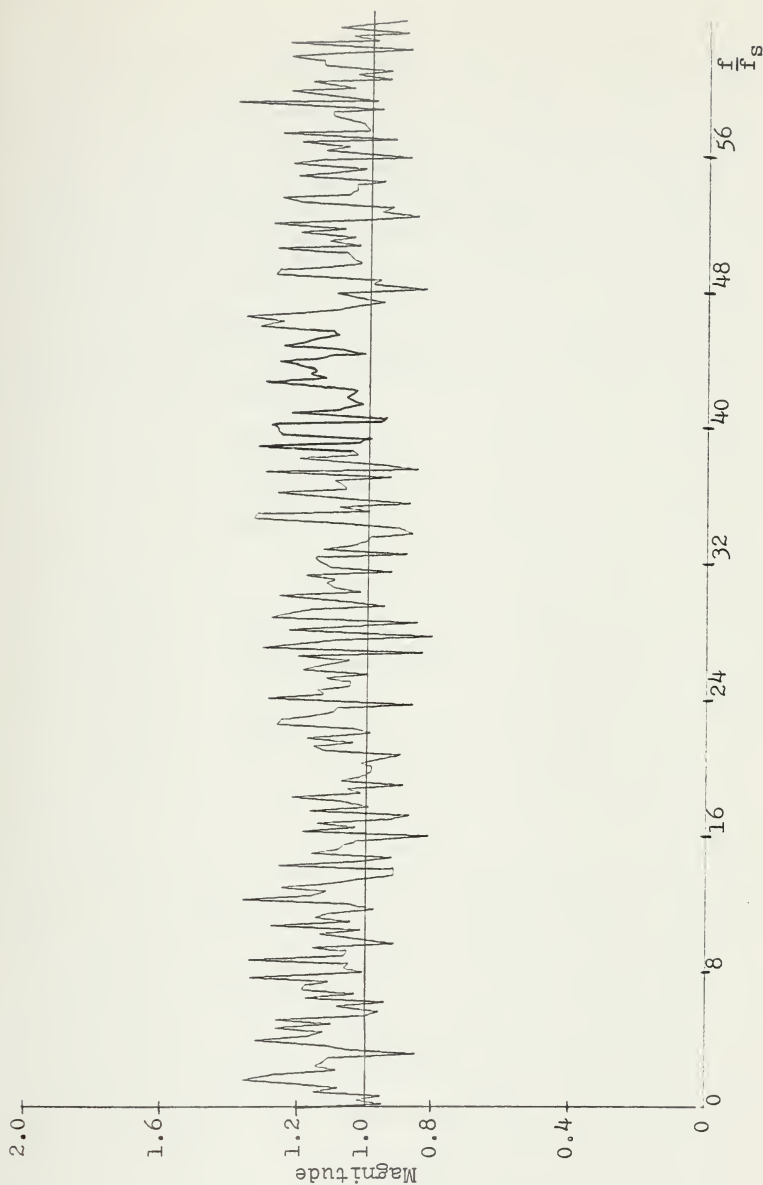


Figure 54 - Magnitude Plot of Input Impulse,  $p=30\%$ .



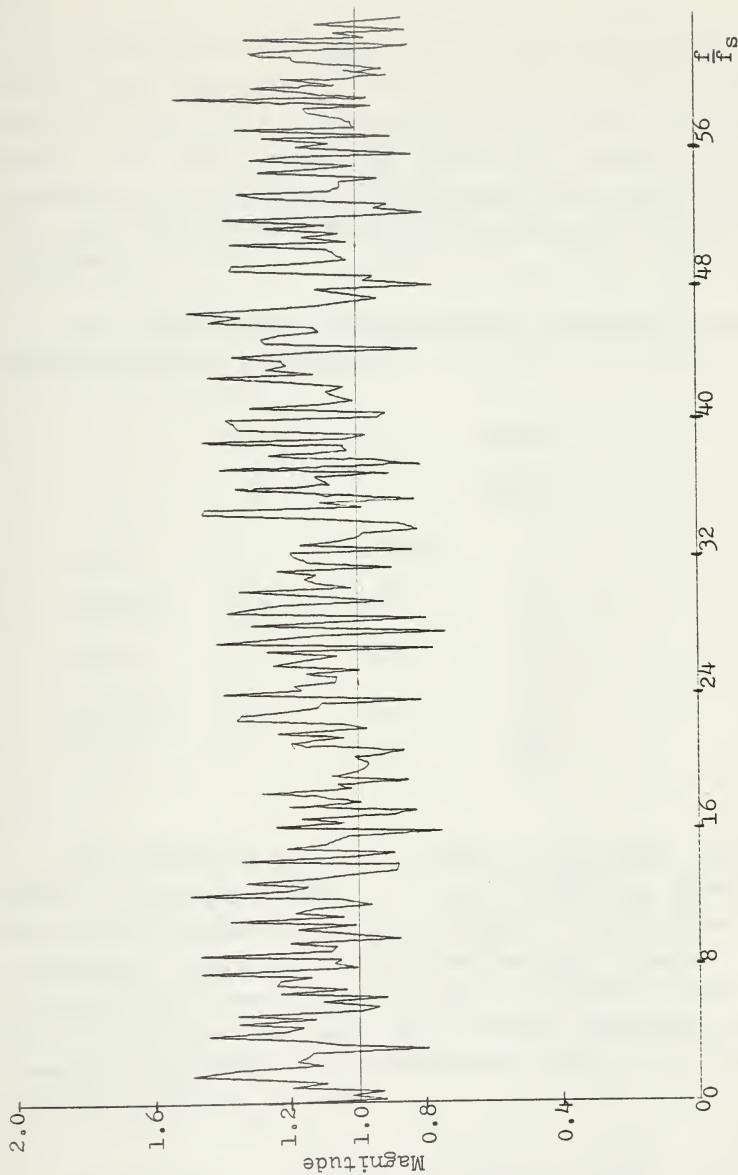


Figure 55 - Magnitude Plot of Input Impulse,  $p=40\%$ .



Fig.[57] was the second waveform used for computing the DFT. The magnitude of the one-sided spectra is shown in Figs.[58] thru [63]. Degradation of the output spectrum was never so severe that the  $(\sin x)/x$  function was not recognizable, but as pointed out above, the average value of the magnitude shows a moderate increase as the value of "p" increases.

The third input waveform, shown in Fig.[64], consisted of the following components.

	Ampl.	$f/f_s$	Phase Shift
sine	1.0	14.125	0°
ccsine	-0.6	15.0	0°
sine	1.0	16.0	0°
cosine	0.8	16.25	0°
sine	-0.4	17.5	60°

The normalized one-sided spectra and phase plots are shown in Fig.[65] thru [76]. In Fig.[65], the spreading of the 14.125 component is due to leakage. the 16.0 and 16.25 components appear to be indistinguishable, however, this is a miss-leading result caused by the continuous curve plot. If a point graph had been used, the two components would be readily identified, an expected result since the frequency resolution is  $f/f_s = 0.25$ .

Examination of the phase plots reveals that the degradation is more severe than that of the magnitude plots,



even for small values of "p". As in the previous examples, the magnitude plots also exhibit a general increase in the component magnitude except for the  $f/f_s = 16.0$  component. This component shows a very distinguishing decrease as "p" increases. By re-examining the magnitude plots of the impulse waveform, it is obvious that the  $f/f_s = 16.0$  component will show a decrease as "p" increases. These plots also show that a more severe degradation of the one-sided spectra would result for a similar set of components near  $f/f_s = 48.0$ .





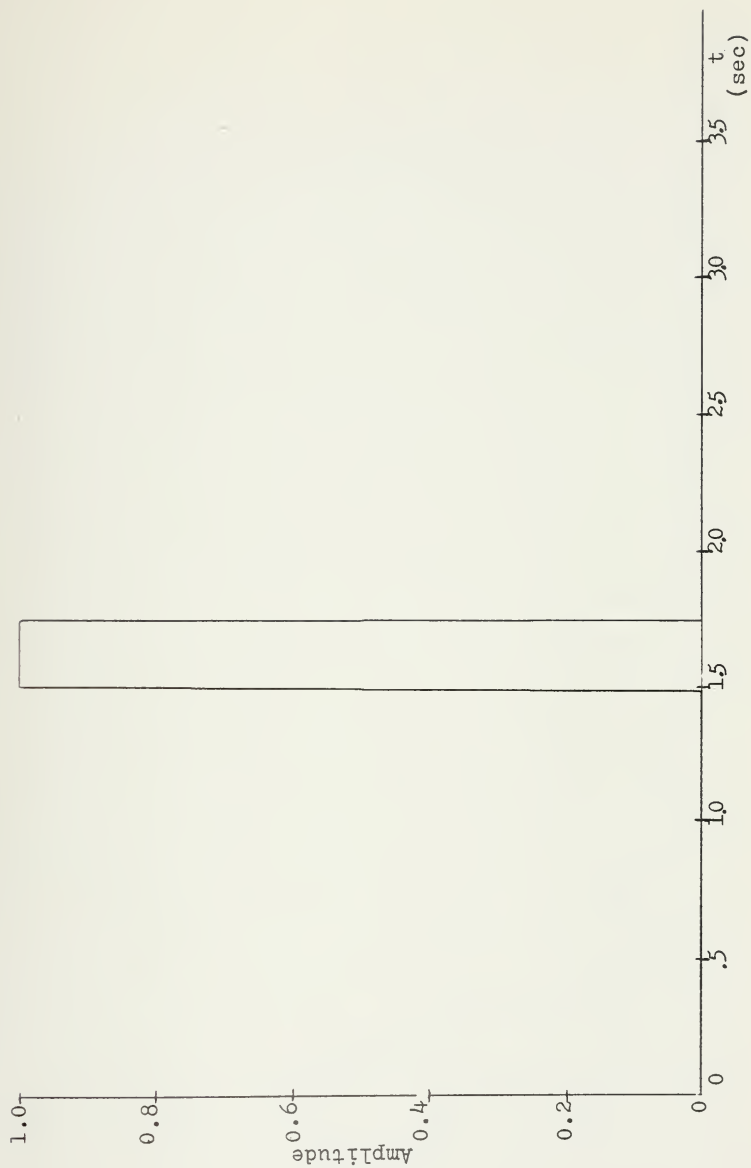


Figure 56 - Rectangular Pulse Input.



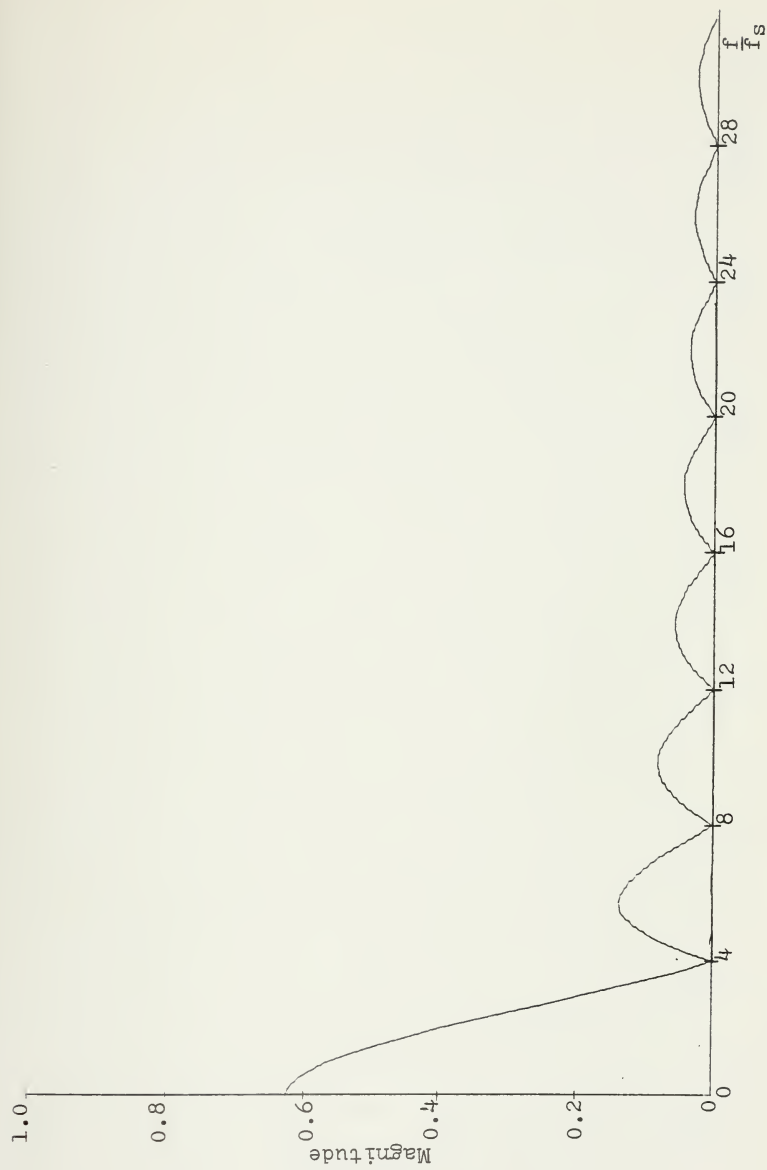


Figure 57 - Magnitude Plot of Rectangular Pulse Input,  $p=0\%$ .



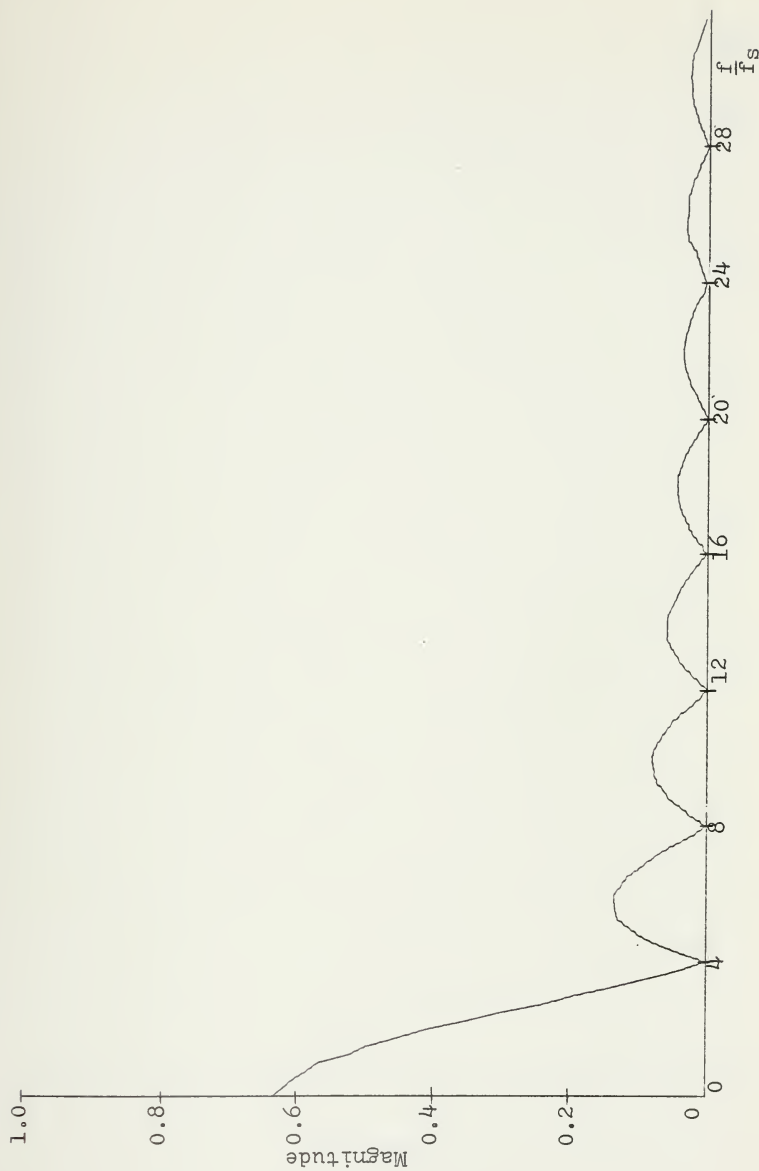


Figure 58 - Magnitude Plot of Rectangular Pulse Input,  $p=4\%$ .



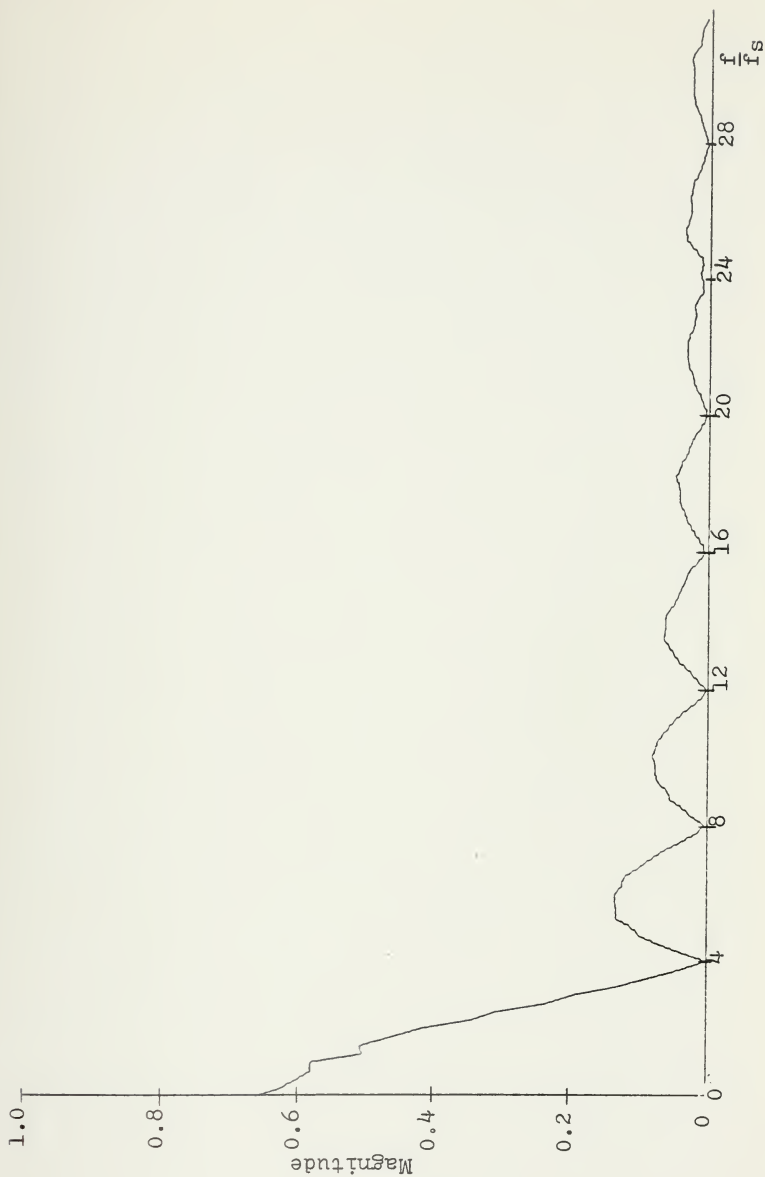


Figure 59 - Magnitude Plot of Rectangular Pulse Input,  $p=10\%$ .





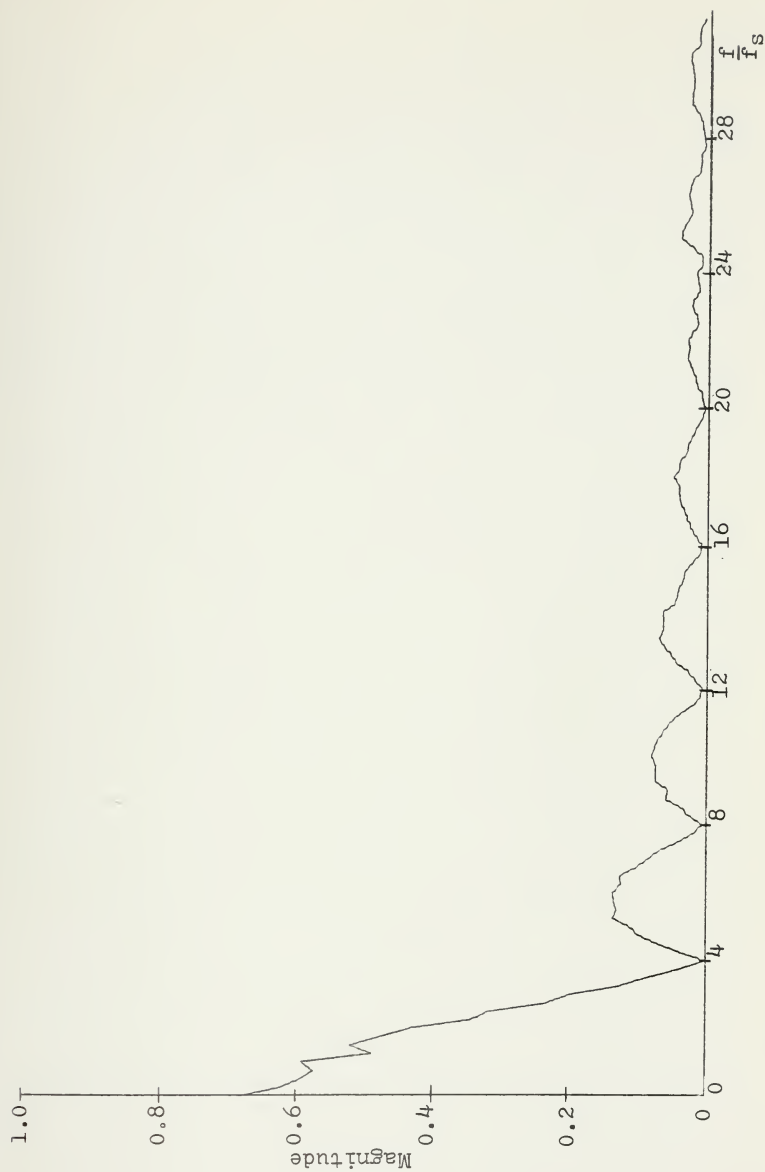


Figure 60 - Magnitude Plot of Rectangular Pulse Input,  $p=20\%$ .



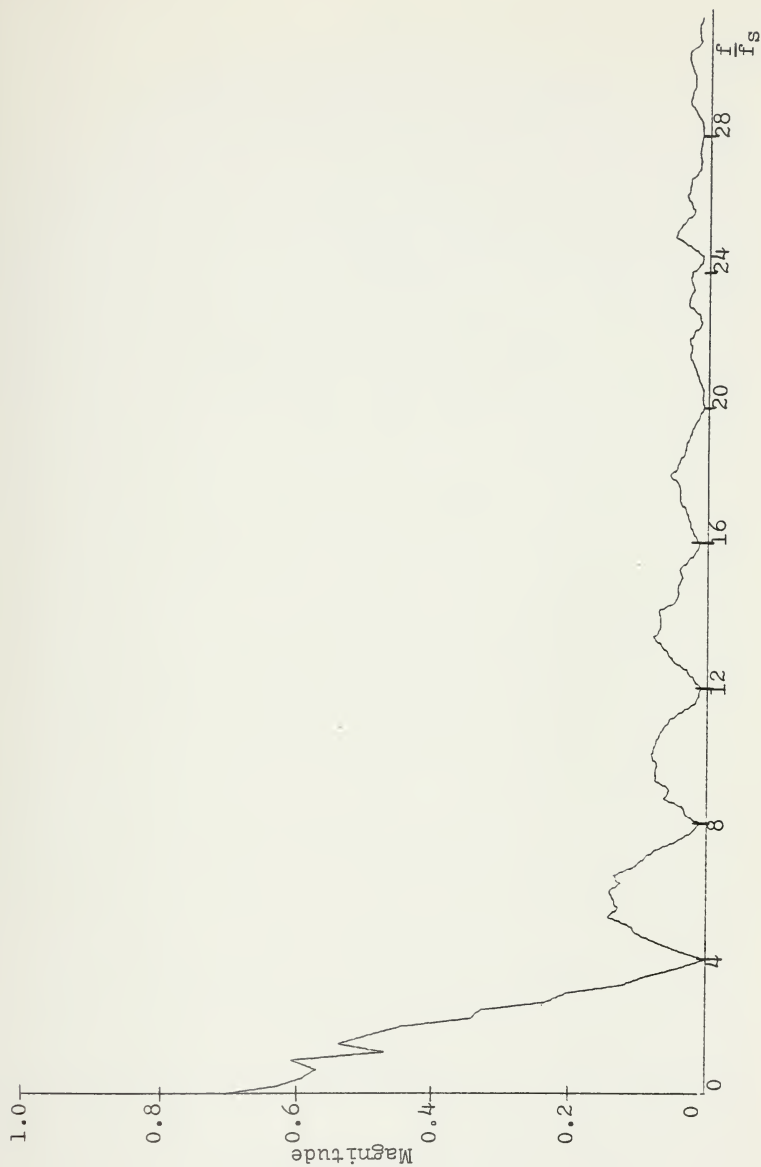


Figure 61 - Magnitude Plot of Rectangular Pulse Input,  $p=30\%$ .





Figure 62 - Magnitude Plot of Rectangular Pulse Input,  $p=40\%$ .



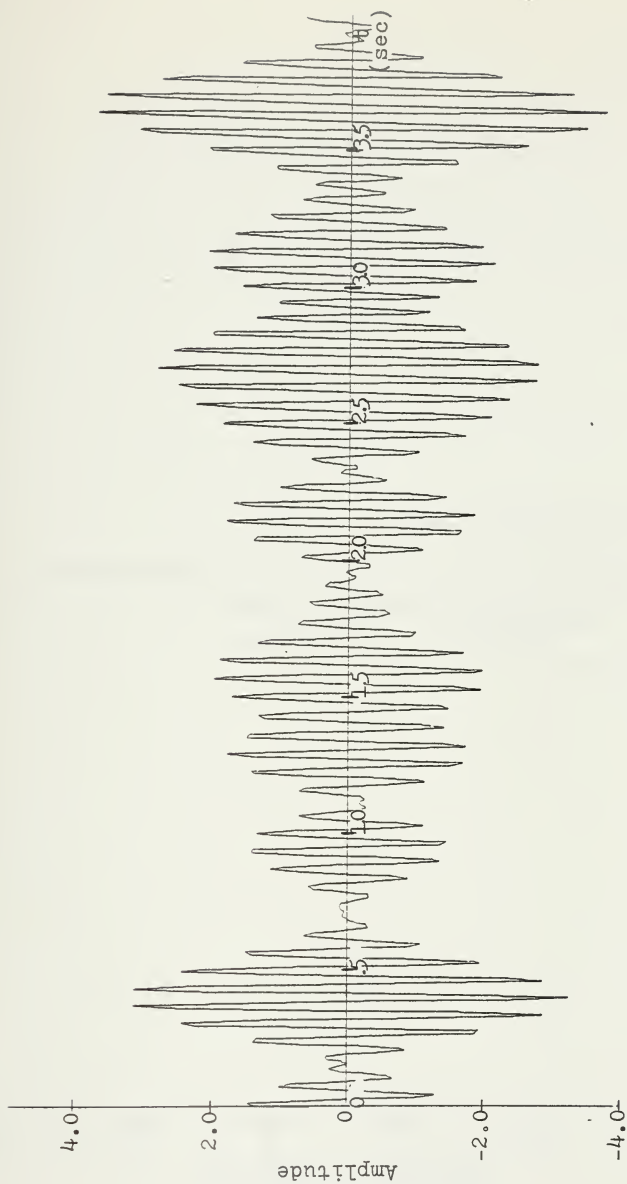


Figure 63 - Sinusoidal Input.





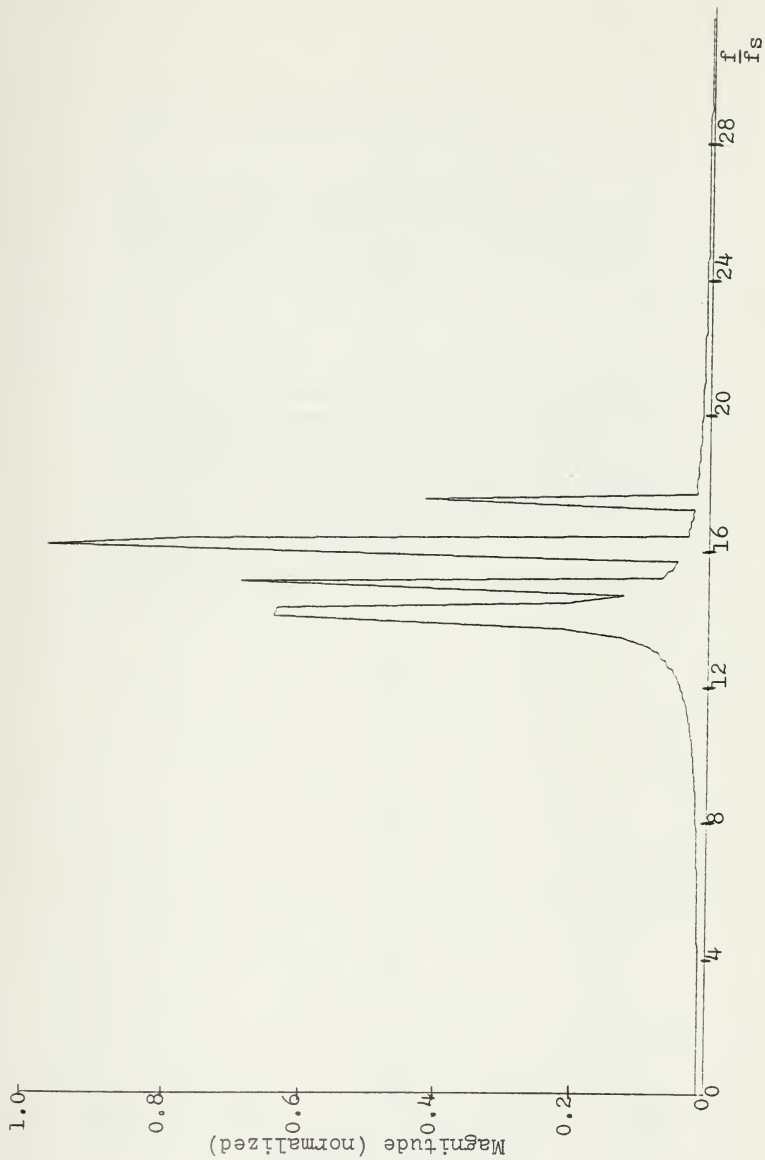


Figure 64 - Magnitude Plot of the Sinusoidal Input,  $p=0\%$ .



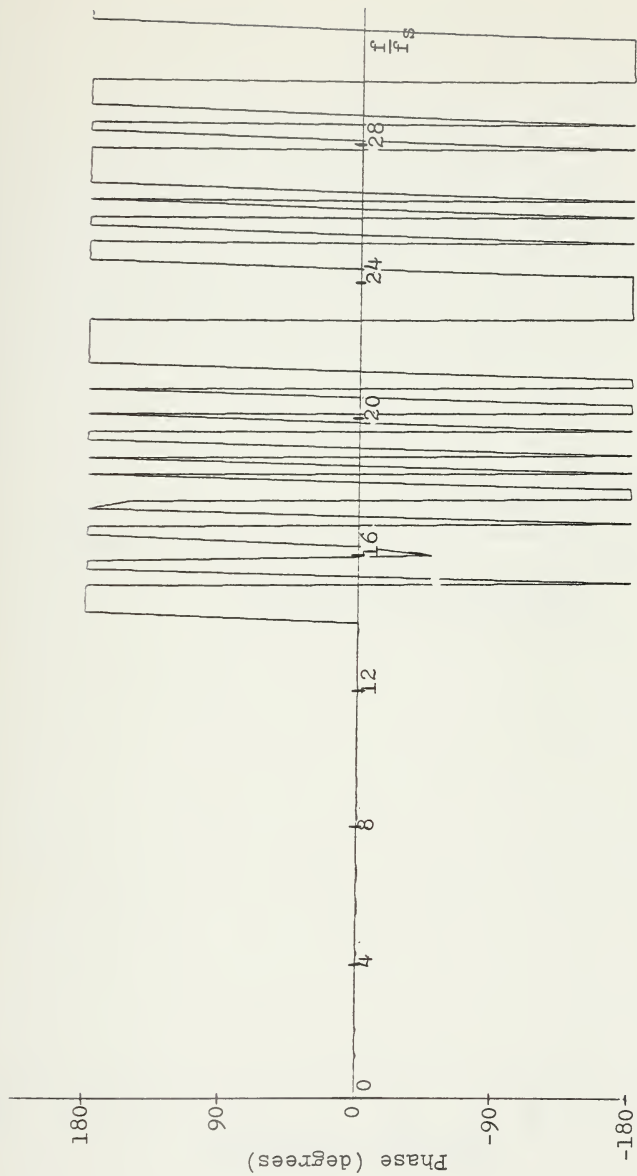


Figure 65 - Phase Plot of the Sinusoidal Input,  $p=0\%$ .



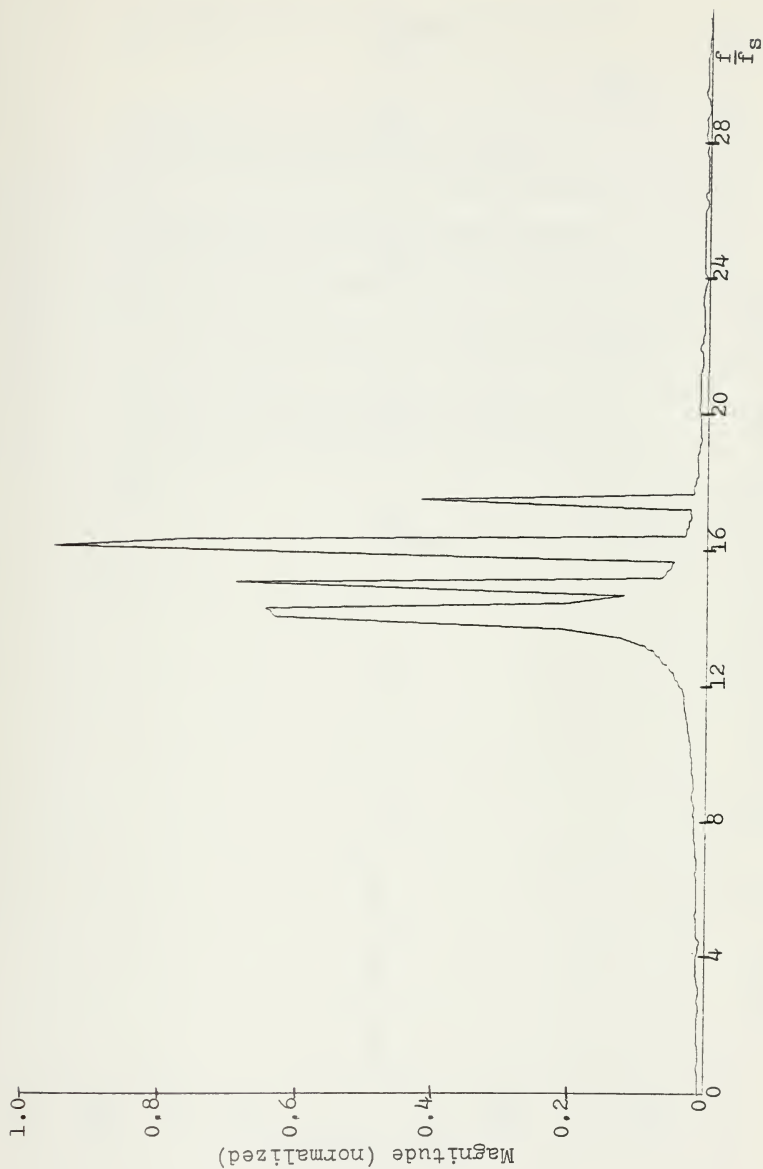


Figure 66 - Magnitude Plot of the Sinusoidal Input,  $p=4\%$ .



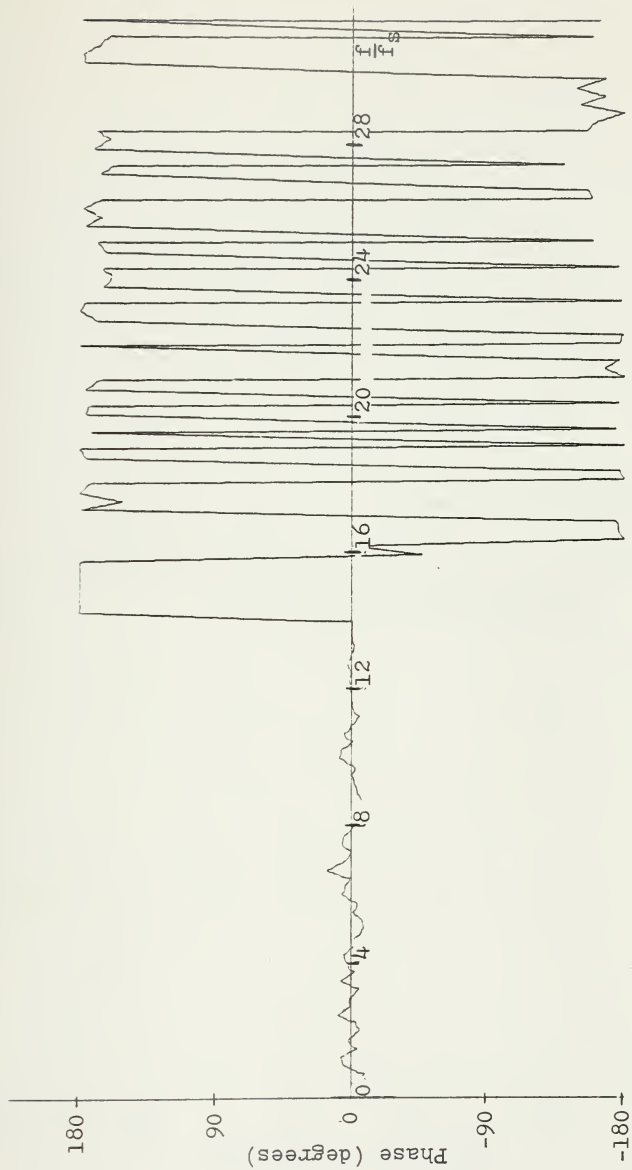


Figure 67 - Phase Plot of the Sinusoidal Input,  $p=4\%$ .





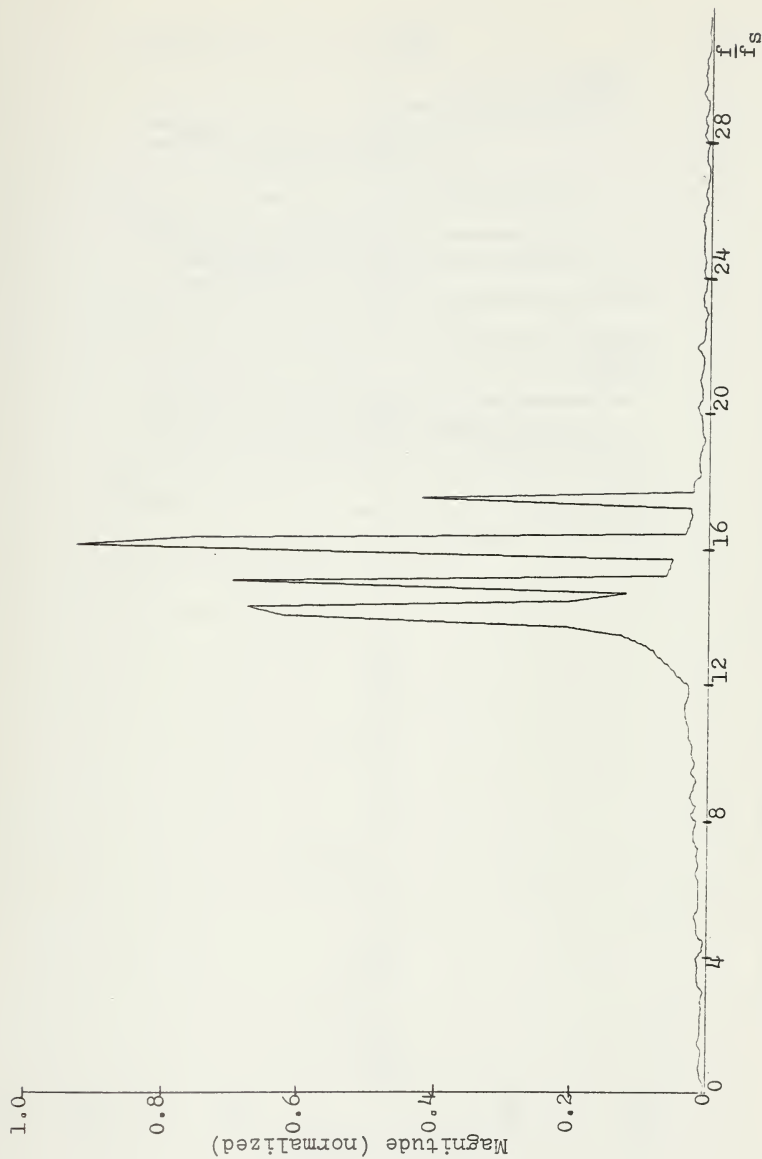


Figure 68 - Magnitude Plot of the Sinusoidal Input,  $p=10\%$ .



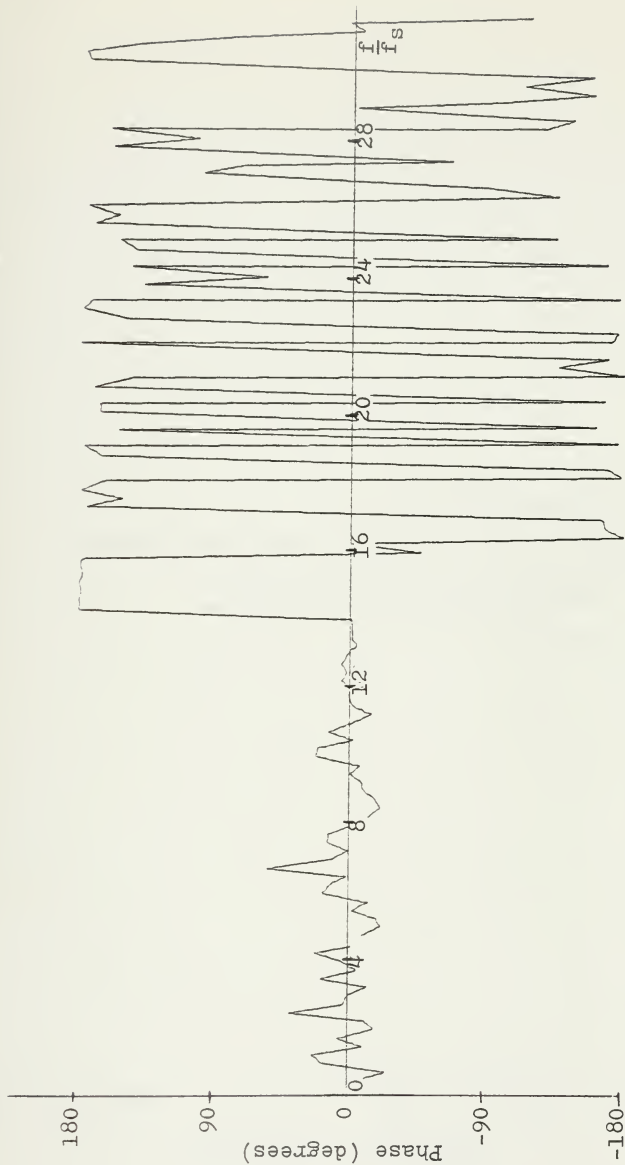


Figure 69 - Phase Plot of the Sinusoidal Input,  $p=10\%$ .



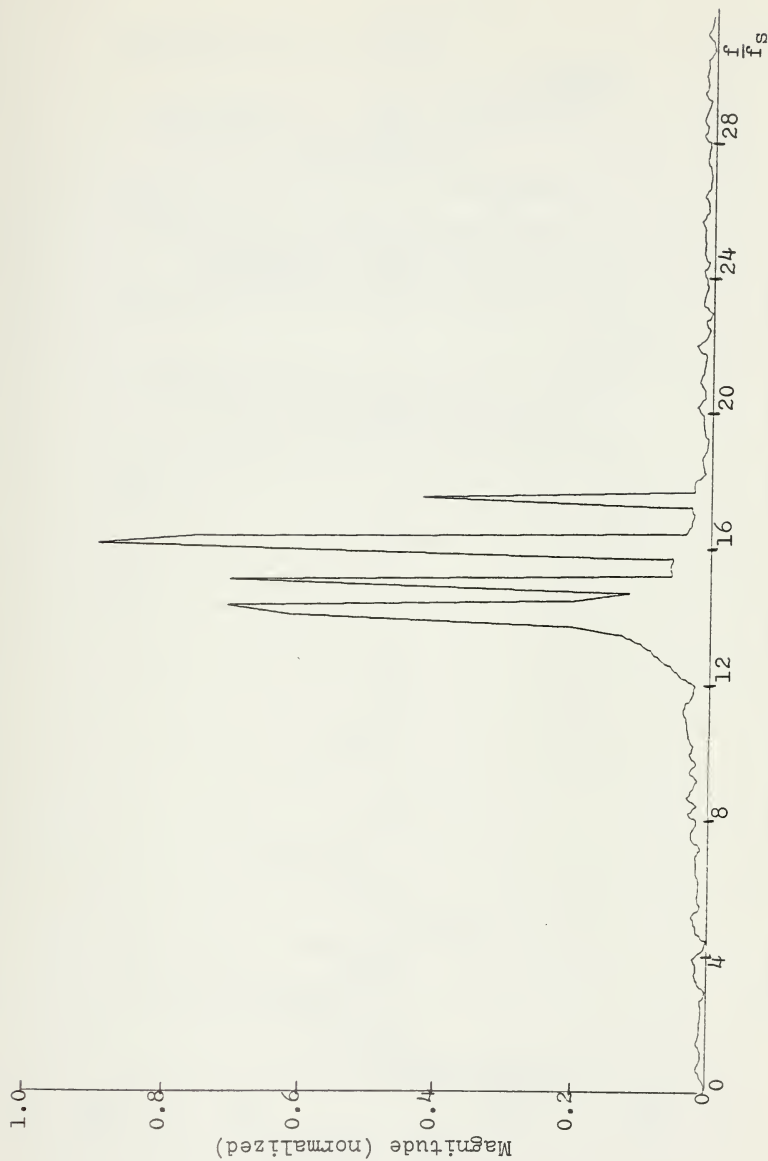


Figure 70 - Magnitude Plot of the Sinusoidal Input,  $p=20\%$ .



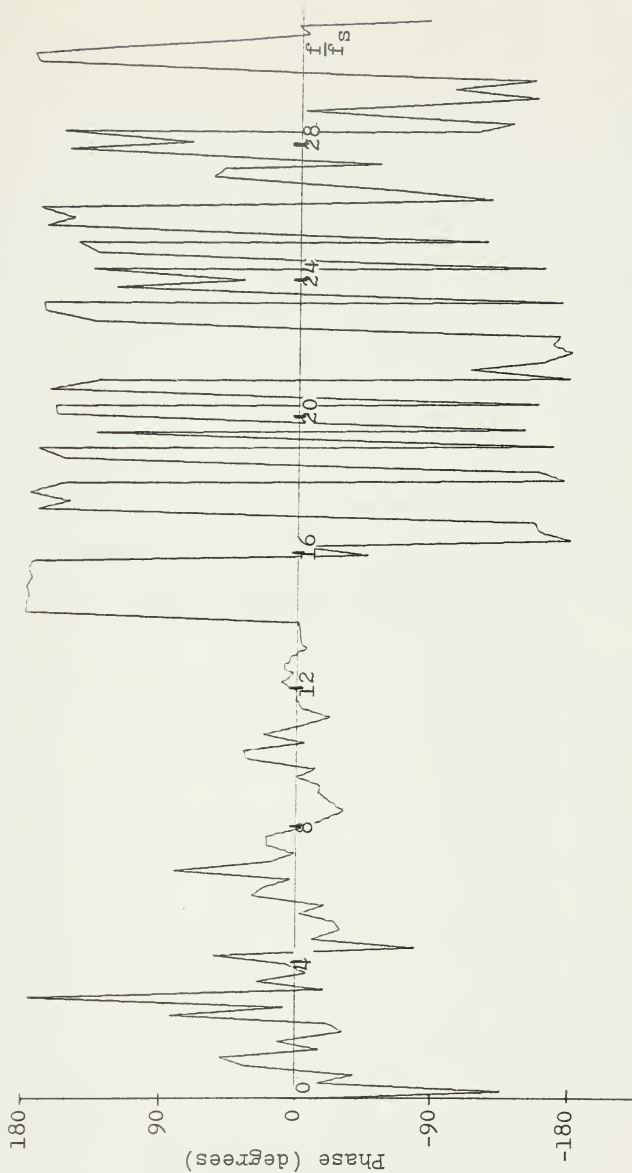


Figure 71 - Phase Plot of the Sinusoidal Input,  $p=20\%$ .





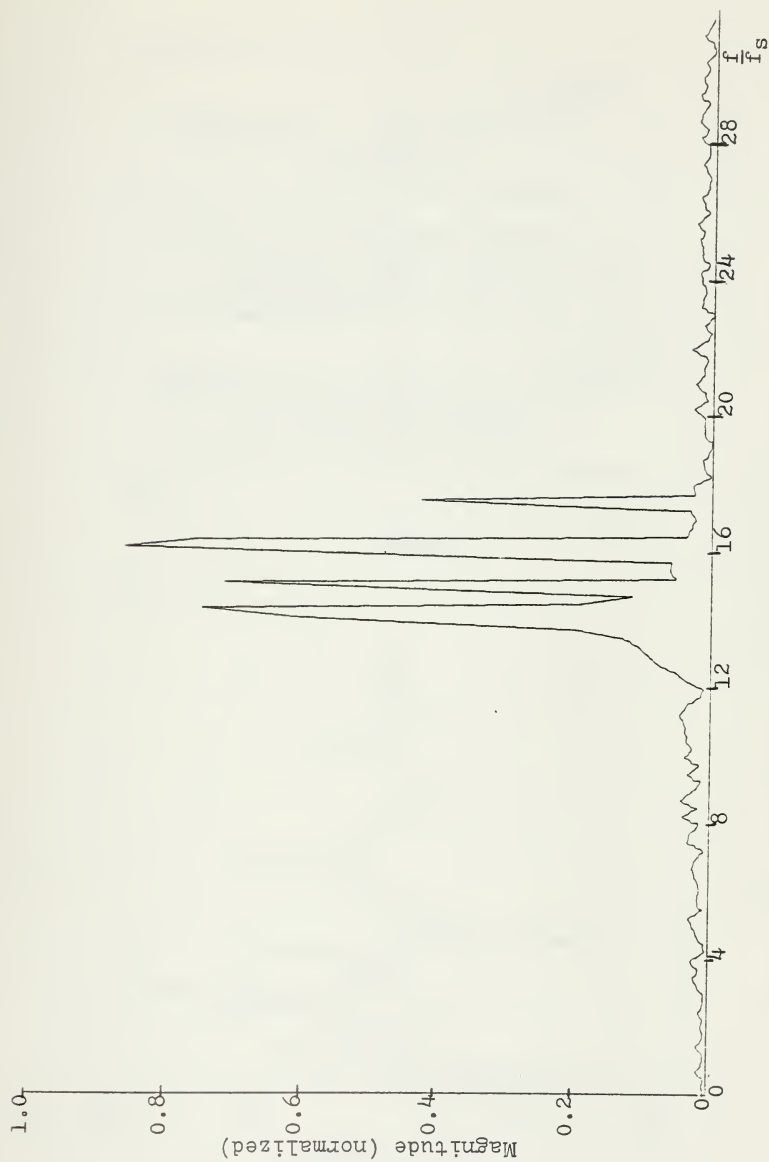


Figure 72 - Magnitude Plot of the Sinusoidal Input,  $p=30\%$ .



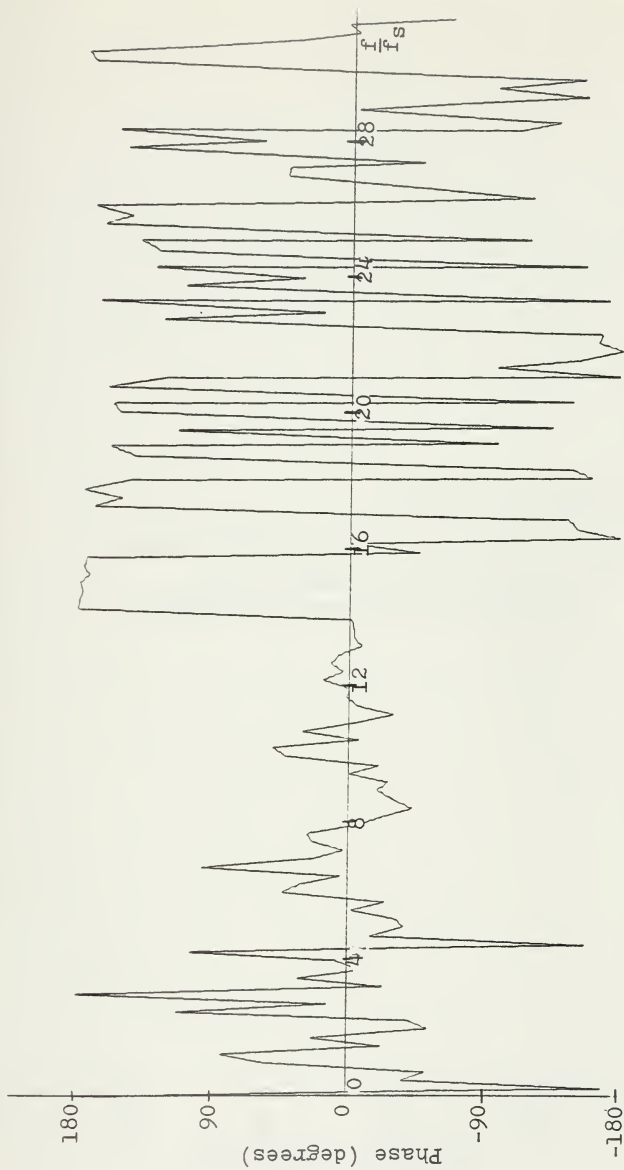


Figure 73 - Phase Plot of the Sinusoidal Input,  $p=30\%$ .



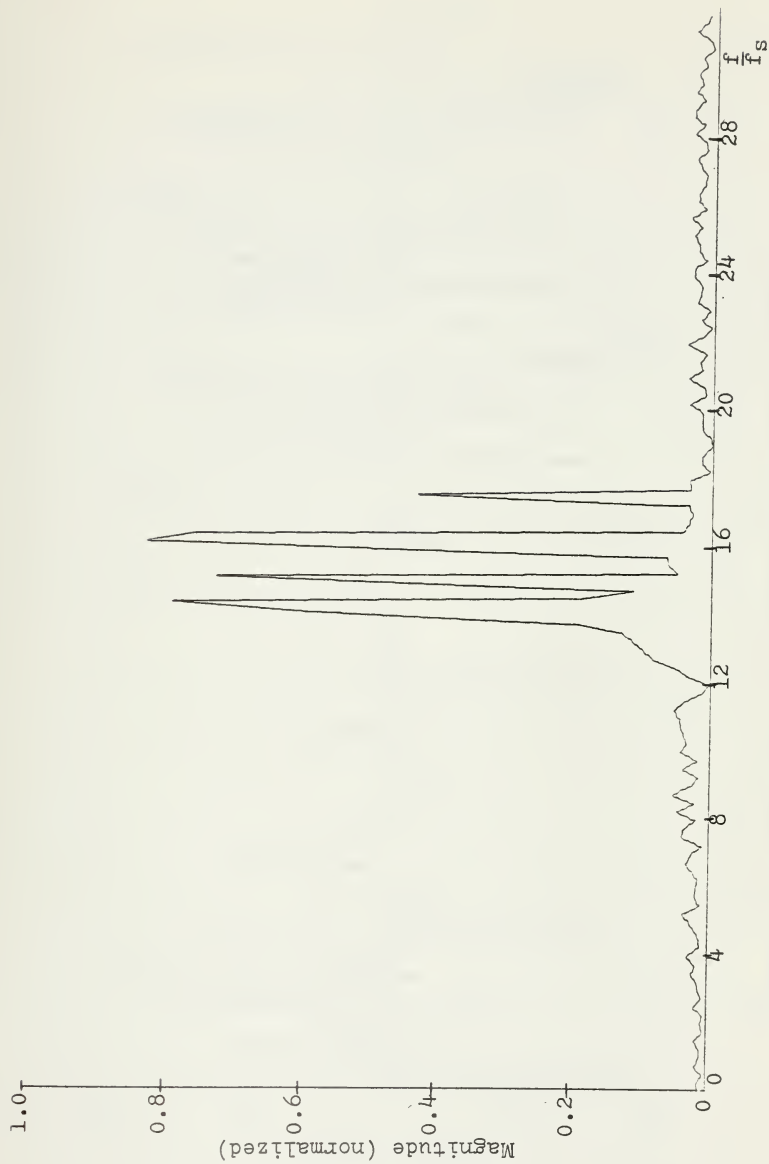


Figure 74 - Magnitude Plot of the Sinusoidal Input,  $p=40\%$ .



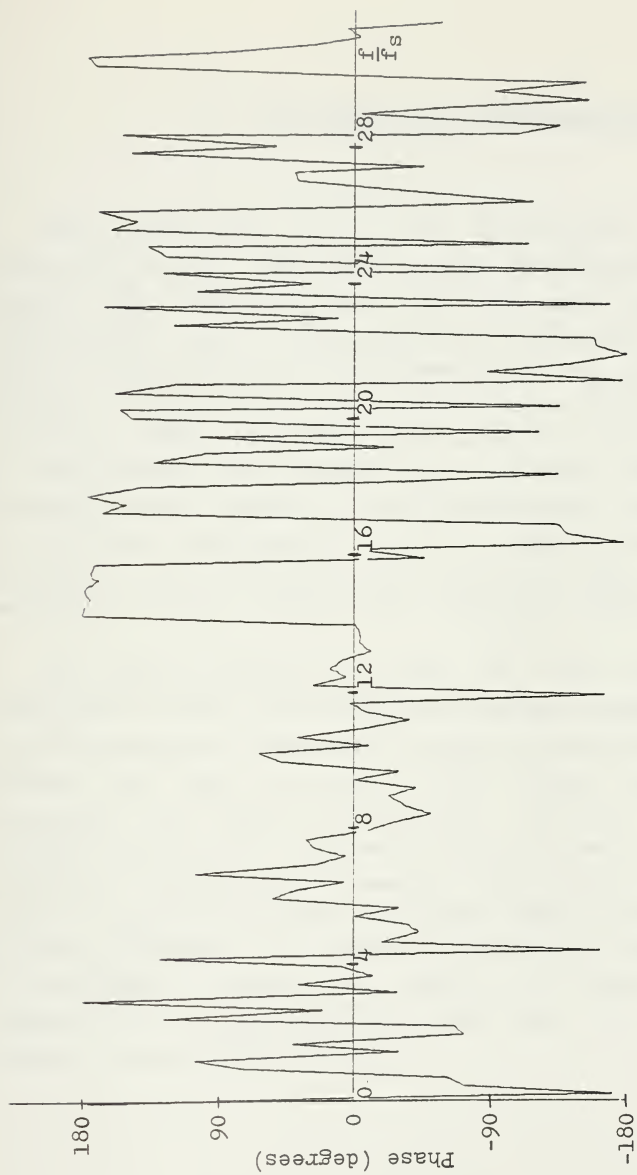


Figure 75 - Phase Plot of the Sinusoidal Input,  $p=40\%$ .





## VI. CONCLUSIONS AND RECOMMENDATIONS

The FFT algorithm is a well established method for computing the DFT. The chirp-z-transform algorithm can also compute the DFT and since the bulk of the required computations can be performed by a transversal filter, it lends itself naturally to implementation with sampled analog devices. The prime transform algorithm has demonstrated its ability to compute the DFT and like the CZT algorithm, can be implemented with sampled analog devices. The fact that the multipliers required by the CZT algorithm are replaced by permuting memories in the prime transform algorithm may or may not be an advantage. It will depend upon the speed and dynamic range required for a particular application.

This paper evaluated the errors in the weights of a transversal filter for weighting provided externally to the CCL by resistors. Preliminary results indicate that the tap weights may vary as much as 20% before the frequency components are greatly distorted. However, additional analysis needs to be conducted before conclusive results are presented. In addition to resistor weighting, a split-gate weight technique has been developed which weights the taps of the transversal filter during fabrication. For this type of weighting, quantization errors are introduced. This suggests that a different error model will be required to conduct a sensitivity analysis.



## APPENDIX A

### ALGORITHM TEST DATA

Several test waveforms were computer generated to evaluate the performance of the varicus algorithms used to compute the IFT.

Case 1-→ This waveform ccnsisted of the following ccnpnents:

	Amp.	$f/f_s$	Phase Shift
D-C	0.8	0.0	0°
cosine	1.0	2.0	0°
sine	2.0	6.0	0°
ccsine	1.0	14.0	45°

The intert of this case was to see how well the algorithms identified the magnitude and phase information of each frequency ccnpnent.



Case→ This waveform consisted of the following components

	Amp.	$f/f_s$	Phase Shift
I-C	0.8	0.0	0°
cosine	1.0	2.0	0°
sine	2.0	6.0	0°
sine	-1.5	10.5	0°
ccsine	1.0	14.0	45°

This case illustrates what happens when leakage is present. The data window truncates the  $f/f_s = 10.5$  component at a non-integer multiple of the wavelength, which is the cause of leakage.

Case 3→ This waveform consisted of the following components

	Amp.	$f/f_s$	Phase Shift
I-C	0.8	0.0	0°
ccsine	1.0	2.0	0°
sine	2.0	6.0	0°
ccsine	1.0	14.0	45°
ccsine	-2.0	22.0	0°



This waveform has a component greater than  $(f_s)/2$ , thus this case was used to illustrate the aliasing phenomenon and the effect  $N$  (being odd or even) had on the folding of this component into the analyzing bandwidth.





## APPENDIX B

### BRIEF REVIEW OF INTEGERS AND INDICES

The following definition and theorems are provided for a better understanding of the derivation of the prime transform algorithm.

#### A. DEFINITIONS

Unit→ a unit is an integer that divides every integer. Since +1 and -1 divide every integer, they are both units.

Null element→ it is an integer that divides only itself. No integer different from zero is a null element. Zero is the null element of the rational integers.

Associates of an integer→ they are the results of multiplying the integer by the units.

Prime→ it is an integer, not a unit, that is divisible by only its associates and the units. This definition implies that the greatest common divisor of a prime "p" and an integer "a" is 1, or the positive associate of "p".

Composite→ it is an integer that is not the null element, a unit, or a prime, e.g. the integer 256 is said to be highly composite while the integer 563 is moderately composite.



Relatively prime  $\rightarrow$  two or more integers are prime to each other or relatively prime, if their greatest common divisor is +1. The integers 6, -9, 14 are relatively prime.

Totient of  $m \rightarrow$  denoted  $\phi(m)$  is the number of positive integers less than " $m$ " and relatively prime to " $m$ ".

## E. THEOREMS

1. Two integers " $a$ " and " $b$ " are said to be congruent modulo and integer " $m$ " if " $m$ " is an integer divisor of  $a-b$ . This relation between " $a$ " and " $b$ " is most often denoted by

$$a \equiv b \pmod{m} \quad (E-1)$$

and is read " $a$  is congruent to  $b \pmod{m}$ ". But for the ease of notation, it is denoted as

$$(a) = b \pmod{m} \quad (E-2)$$

in the derivation of the prime transform algorithm.

If " $a$ " is congruent to " $b$ " modulo ( $m$ ) and  $0 \leq b < m$ , " $b$ " is called the residue of  $a$  modulo ( $m$ ). If " $m$ " is not an integer divisor of  $a-b$ , " $a$ " and " $b$ " are said to be distinct modulo  $m$ .

2. If " $a$ " and " $m$ " ( $\neq 0$ ) are relatively prime integers, then

$$(a^{\phi(m)}) = 1 \pmod{m} \quad (E-3)$$



Corollary of 2. If "p" is a prime which is not a divisor of an integer "a", then

$$((a^{p-1})) = 1 \text{ mod } (p) \quad (E-4)$$

If "a" and "u" (>0) are relatively prime integers, and "u" is the least positive integer for which

$$((a^u)) = 1 \text{ mod } (u) \quad (E-5)$$

"u" is called the exponent to which "a" belongs modulo (m). If the exponent to which "a" belongs modulo (m) is  $\phi(m)$ , "a" is defined as a primitive root modulo (m) (or a primitive root of m).

3. The only integers which possess primitive roots are  $2, 4, p^2, 2p^r$ , where "p" is an odd prime. If "m" does have a primitive root, it has  $\phi(\phi(m))$  distinct roots modulo (m).

Let "p" be any prime, and let "r" be any primitive root of "p". To each integer "a" relatively prime to "p" there corresponds a unique integer "i" such that

$$((a)) = r^i \text{ mod } (p) \quad (0 < i < p-1) \quad (E-6)$$

The integer "i" is called the index to the base "r" of a modulo (p). This is written as

$$i = \text{ind}_r a \quad (E-7)$$



# Properties of Indices

$$(a) \text{ ind}_r 1 = 0 \quad (E-8)$$

$$(b) \text{ ind}_r (-1) = (p-1)/2 \quad (E-9)$$

$$(c) \text{ ind}_r (ab) \equiv \text{ind}_r a + \text{ind}_r b \pmod{p-1} \quad (E-10)$$

$$(d) \text{ ind}_r a^n \equiv n \text{ ind}_r a \pmod{p-1} \quad (E-11)$$

The power residue of an integer "b" to the base "r" is simply the integer "s" such that

$$(s) = r^b \pmod{p} \quad (0 < s < p) \quad (E-12)$$

Since the power residue of  $\text{ind}_r a$  is congruent to a modulo (p), the concept of indices and power residues is analogous to that of logarithms and antilogarithms.





# APPENDIX C

## PROGRAM LISTING OF THE FFT ALGORITHM

```

C THIS PROGRAM USES THE DECIMATION-IN-TIME (COOLEY-TUKEY)
C FFT ALGORITHM TO COMPUTE THE DISCRETE FOURIER TRANSFORM
C (DFT) COEFFICIENTS OF A COMPLEX INPUT.
C
C DIMENSION TV(1024)
C DIMENSION FV(514)
C DIMENSION Y(1024)
C DIMENSION AMP(1024), PHASE(600)
C DIMENSION S(1012), INV(1012), M(3)
C DIMENSION TIME(600), XS(600)
C COMPLEX Z
C COMPLEX*8 A(1024,1,1)
C INTEGER*4 ITB(12)/12*0/
C REAL*4 RTB(28)/28*0.0/
C
C READ IN THE POWER OF TWO "KK" WHICH DETERMINES THE NUM-
C BER OF DATA POINTS AND COMPLEX FREQUENCY COMPONENTS AND
C THE LENGTH OF THE DATA WINDOW "T" IN SECONDS.
C
C READ(5,6) KK, T
C FORMAT(15,E10.7)
C N1=2**KK
C N2=1
C N3=1
C NF=N1/2
C TS = T/N1
C
C COMPUTE THE SAMPLED VALUES OF AN ARBITRARY REAL WAVEFORM.
C
C DO 215 J = 1, N1
C   K = J-1
C   XS(J) = 0.8 + COS(6.28318 * 2. * K * TS) + (2.0)*SIN(6.
C 428318 * 6.0 * K * TS) +
C 428318 * COS((6.28318 * 14. * K * TS) + 0.78539)
C 428318 * (2.0)*COS(6.28318 * 22.0 * K * TS)
C 215 CONTINUE
C   M(1)=KK
C   M(2)=0
C   M(3)=0
C
C CALCULATE THE FUNDAMENTAL FREQUENCY AND ITS HARMONICS.
C CALCULATE THE TIME AXIS VALUES.
C
C FUND = (1./T)
C FV(1)=0.0
C DO 20 J=2, NF
C   FV(J)=FV(J-1)+FUND
C 20 CONTINUE
C TV(1)=0.0
C DO 30 K=2, N1
C   TV(K)=TV(K-1)+TS
C 30 CONTINUE
C

```



```

C   SET THE COMPLEX "A" MATRIX TO ZERO.
C
      CC 2 I1=1,N1
      CC 2 I2=1,N2
      CC 2 I3=1,N3
2  Z (I1,I2,I3)=(0.0,0.0)
      CC 3 I1=1,N1
      CC 3 I2=1,N2
      CC 3 I3=1,N3

C   TRANSFER THE SAMPLED INPUT DATA TO THE COMPLEX "A" MA-
C   TRIX.
      A(I1,1,1) = XS(I1)
      = CCNTINUE

C   COMPUTE THE DISCRETE FOURIER TRANSFORM OF THE SAMPLED
C   INPUT DATA.
      CALL FARM(A,M,INV,S,-1,IFERR)

C   CALCULATE THE MAGNITUDE AND PHASE OF THE FREQUENCY COM-
C   PONENTS.
      CC 33 I1=1,NH
      Z = A(I1,1,1)
      ZREAL = REAL(Z)
      ZIMAG = AIMAG(Z)
      IF (ABS(ZREAL).LT.0.001) ZREAL = 0.0
      IF (ABS(ZIMAG).LT.0.001) ZIMAG = 0.0
      AMF(I1) = CABS(Z)
      PHASE(I1) = 100.0
      IF (ZREAL.EC.0.0.AND.ZIMAG.GE.0.0) PHASE(I1) = 90.0
      IF (ZREAL.EC.0.0.AND.ZIMAG.LT.0.0) PHASE(I1) = -90.0
      IF (ZREAL.GE.0.0.AND.ZIMAG.EC.0.0) PHASE(I1) = 0.0
      IF (ZREAL.LT.0.0.AND.ZIMAG.EC.0.0) PHASE(I1) = 180.0
      IF (PHASE(I1).NE.100.0) GC TC 33
      PHASE(I1) = (ATAN2(ZIMAG,ZREAL)) * 57.2957
33  CCNTINUE

C   CLTPLOT THE COMPUTED DATA.
C
      WRITE(6,47) T,TS,N1,FLNC
47  FORMAT(/,' SAMPLE TIME = ',F7.4,' SECONDS'//,
      &' SAMPLE INTERVAL = ',F6.4,' SECONDS'//,
      &' NUMBER OF SAMPLES = ',I4//,
      &' FREQUENTIAL FREQUENCY = ',F6.1,' HZ')
      ITE(3) = 8
      ITE(4) = 5
      ITE(12)=1
      NH = N1
      RTE(1) = 0.1
      RTE(2) = 2.0

C   PLOT THE SAMPLED INPUT WAVEFORM.
C
      CALL CRAWP(NH,TV,XS,ITE,RTE)
      ITE(2) = 3
      NH = N1/2
      RTE(1) = 2.0
      RTE(2) = 0.2

C   PLOT THE MAGNITUDE OF THE ONE-SIDED SPECTRA.
C
      CALL CRAWP(NH,FV,AMF,ITE,RTE)
      RTE(2) = 90.0

C   PLOT THE PHASE OF THE ONE-SIDED SPECTRA.
C
      CALL CRAWP(NH,FV,PHASE,ITE,RTE)
      STOP
      END

```



# APPENDIX C

## PROGRAM LISTING OF THE CZT ALGORITHM

```

C THIS PROGRAM COMPUTES THE DISCRETE FREQUENCY COMPONENTS
C OF AN ARBITRARY COMPLEX INPUT WAVEFORM BY THE CHIRP-Z-
C TRANSFORM ALGORITHM.

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)
COMMON WI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),SS(512),F,T,A,RN,SPL,PRRW,TRW,PCRW
COMMON FCT1,PCT2,PCT3,PCT4,PCT5,PCT6
INTEGER*4 ITB(12)/12*0/
REAL*4 RTB(28)/28*0.0/

READ THE PARAMETER "I1", WHICH DETERMINES IF DATA IS TO
BE READ IN OR CALCULATED WITHIN THE PROGRAM. IF "I1" IS
EQUAL TO "0", THE PROGRAM WILL GENERATE A COMPLEX SIN-
ECICAL INPUT FOR A SINGLE FREQUENCY. IF "I1" IS NOT
EQUAL TO ZERO, AN ARBITRARY REAL INPUT MAY BE READ IN AND
THE IMAGINARY PART (YS) IS SET TO ZERO.

      IF(I1) = 1
      READ(5,110) I1
110  FORMAT(I5)

      "N" IS THE NUMBER OF TIME AND FREQUENCY SAMPLE POINTS.
      "F" IS THE FREQUENCY TO BE USED IN SUBROUTINE WAVE AND
      "T" IS THE LENGTH OF THE INPUT DATA WINDOW IN SECONDS.
      "SPL" DETERMINES THE RATE OF SPIRALING OF THE CANTOUR
      IN THE Z-PLANE. IF "SPL" IS EQUAL TO ZERO, THE CANTOUR
      IS A UNIT CIRCLE IN THE Z-PLANE WHICH CORRESPONDS TO
      CALCULATING THE DISCRETE FOURIER TRANSFORM.

      READ(5,105) F,T,SPL,N
105  FORMAT(E10.7,I5)

      READ IN THE PARAMETER "PRRW" WHICH DETERMINES WHETHER OR
      NOT RANDOM VALUED PRE-MULTIPLYING COEFFICIENTS WILL BE
      USED IN THE COMPUTATION. PARAMETER "TRW" DETERMINES
      WHETHER OR NOT RANDOM VALUED TAPPING WEIGHTS IN THE CON-
      VOLUTION FILTERS WILL BE USED IN THE COMPUTATION.
      PARAMETER "PCRW" DETERMINES WHETHER OR NOT RANDOM VALUED
      POST-MULTIPLYING WEIGHTS WILL BE USED IN THE COMPUTATION.

      READ(5,101) PRRW,TRW,PCRW
101  FORMAT(E10.7)

      "PCT1" AND "PCT2" DETERMINE THE RANGE OF VALUES THE
      PRE-MULTIPLYING COEFFICIENTS CAN DEVIATE FROM A NOMINAL
      VALUE. "PCT3" AND "PCT4" DETERMINE THE RANGE OF VALUES
      THE FILTER TAPPING WEIGHTS CAN DEVIATE FROM A NOMINAL
      VALUE. "PCT5" AND "PCT6" DETERMINE THE RANGE OF VALUES
      POST-MULTIPLYING COEFFICIENTS CAN DEVIATE FROM A NOMINAL
      VALUE.

```



```

102 READ(5,102) PCT1, PCT2, PCT3, PCT4, PCT5, PCT6
   FCFMAT(6E10.3)
   FREQ(1) = 0.0
   FN = N
   IF(11.5C.0) GO TO 5
C   READ IN THE DATA AND SET "YS" EQUAL TO ZERO
   READ(5,120)(XS(M),M=1,N)
120 FCFMAT(8E10.5)
   CC 40 M = 1,N
   C = M + 1
   YS(M) = 0.0
   FREQ(J) = M * (1./T)
40 CCNTINLE
   GC TC 5

C
C   COMPLETE THE VALUES OF AN ARBITRARY COMPLEX WAVEFORM
C   SAMPLED "N" TIMES.
5   CALL WAVE
5   CALL FRMLT
   CALL XSLM
   CALL TAFW
   CALL CCNV
   CALL PCMULT
   CALL AMFPA

C
C   CLTPLT THE COMPUTED DATA.
C   WRITE(6,320) IRUN,PCT2
320 FCFMAT(1,2X,'THIS RUN IS NUMBER ',I3,
   S' AND THE PERCENT DEVIATION IS ',F3.2)
   ITE(3) = 8
   ITE(4) = 5
   ITE(12) = 1
   RTE(1) = 0.1
   RTE(2) = 2.0
   NLMPST = N

C
C   FLCT THE SAMPLED INPUT WAVEFORM.
C   CALL CRAWP(NUMPTS,TIME,XS,ITE,RTE)
   RTE(2) = 0.4

C
C   FLCT THE IMPULSE RESPONSE OF THE COSINE FILTER.
C   CALL CRAWP(NLMPST,TIME,X,ITE,RTE)

C
C   FLCT THE IMPULSE RESPONSE OF THE SINE FILTER.
C   CALL CRAWP(NUMPTS,TIME,Y,ITE,RTE)
   NLMPST = N/2
   RTE(1) = 2.0
   RTE(2) = 0.2

C
C   FLCT THE MAGNITUDE OF THE FREQUENCY COMPONENTS.
C   CALL CRAWP(NUMPTS,FREQ,AMP,ITE,RTE)
   RTE(2) = 90.0

C
C   FLCT THE PHASE OF THE FREQUENCY COMPONENTS.
C   CALL CRAWP(NUMPTS,FREQ,PHASE,ITE,RTB)
   STOP
   END

```





# SUBROUTINE WAVE

THIS SUBROUTINE COMPUTES THE VALUES OF AN ARBITRARY  
SELF-GENERATED COMPLEX WAVEFORM FOR A SINGLE FREQUENCY.  
THE FREQUENCY IS SPECIFIED BY THE USER.

```

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)
COMMON HI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),S(512),F,T,N,RN,SPL,FRF,TRW,PCRW
COMMON PCT1,PCT2,PCT3,PCT4,PCT5,PCT6
TS = T/N
CC EQ N = 1,N
K = M - 1
L = M + 1
FREQ(J) = N * (1./T)
XS(M) = COS(6.28318 * F * K * TS)
YS(M) = SIN(6.28318 * F * K * TS)
CONTINUE
RETURN
END

```



# SUBROUTINE PRMULT

THIS SUBROUTINE COMPUTES THE X-AXIS VALUES FOR PLOTTING  
OUTPUT DATA AND COMPUTES THE PRE-MULTIPLICATION COEFFI-  
CIENTS.

```
COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)
COMMON WI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,FORW
COMMON PCT1,PCT2,PCT3,PCT4,PCT5,PCT6
IX = 33333333
CC 10 N = 1,N
K = N - 1
```

COMPUTE THE TIME INCREMENTS FOR PLOTTING THE OUTPUT.

TIME(N) = K \* (T/RN)

COMPUTE THE PRE-MULTIPLICATION COEFFICIENTS.

```
S = (SPL) * (K**2)/2.
CTR = EXP(S)
XW(N) = (CTR) * COS((3.14159*(K**2))/RN)
YW(N) = (CTR) * SIN((3.14159*(K**2))/RN)
```

IF CALLED, THE PRE-MULTIPLICATION COEFFICIENTS ARE  
CALCULATED FOR A UNIFORMLY RANDOM RANGE DETERMINED BY  
THE PERCENTAGE OF DEVIATION FROM A NOMINAL VALUE.

```
IF (PRRW.NE.1.0) GO TO 10
CALL RANDL(IX,IY,YFL)
RNE = YFL
XW(N) = XW(N) - ((PCT1)*XW(N)) + (RND*PCT2*XW(N))
YW(N) = YW(N) - ((PCT1)*YW(N)) + (RND*PCT2*YW(N))
IY = IY
CONTINUE
RETURN
END
```

10



# SLEROUTINE XSUM

C THIS SLEROUTINE MULTIPLIES THE SAMPLED DATA BY THE PRE-  
C MULTIPLICATION COEFFICIENTS AND SUMS THE REAL AND IMAG-  
C INARY COMPONENTS.

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)  
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)  
COMMON WI(512),GR(512),GI(512),AMP(512),TIME(512)  
COMMON PHASE(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,PCRW  
COMMON FCT1,PCT2,PCT3,PCT4,PCT5,PCT6  
CC 16 M = 1,N  
XR1 = XW(M) \* XS(M)  
XR2 = XS(M) \* (-1. \* YW(M))  
YI1 = YS(M) \* (-1. \* YW(M))  
YI2 = YS(M) \* XW(M)

C SUM THE REAL AND IMAGINARY TERMS OF THE PRE-MULTIPLI-  
C CATION PRIOR TO CONVOLVING IN THE CHIRP FILTERS.

WR(M) = XR1 - YI1  
WI(M) = XR2 + YI2  
16 CONTINUE  
FETURN  
END



# SUBROUTINE TAPW

THIS SUBROUTINE COMPUTES THE TAPPING WEIGHTS FOR THE COM-  
PLEX CONVOLUTION FILTERS.

```

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WP(512)
COMMON HI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,PCRW
COMMON PCT1,PCT2,PCT3,PCT4,PCT5,PCT6
RM = N
CC 11 M = 1,N
K = N - M
S = ((SPL)* (K**2)/2.) * (-1.0)
CTR = EXP(S)
X(M) = (CTR) * COS((3.14159*(K**2))/RN)
Y(M) = (CTR) * SIN((3.14159*(K**2))/RN)
11 CONTINUE
NM1 = N - 1
CC 14 K = 1,NM1
J = N - K
M = N + K
X(M) = X(J)
Y(M) = Y(J)
14 CONTINUE

C
C
C IF CALLED, THE TAPPING WEIGHTS ARE CALCULATED FOR A UNI-
C FORMLY RANDOM RANGE DETERMINED BY THE PERCENTAGE OF DEVI-
C ATION FROM A NOMINAL VALUE.
IF (TRW.NE.1.0) RETURN
NRW = 2 * N - 1
IX = 111111111
CC 75 J = 1,NRW
CALL RANDCL(IX,IY,YFL)
RNC1 = YFL
X(J) = X(J) - ((PCT3)*X(J)) + (RNC1*PCT4*X(J))
Y(J) = Y(J) - ((PCT3)*Y(J)) + (RNC1*PCT4*Y(J))
J = IY
75 CONTINUE
RETURN
END

```





# SIERGLTINE CCNV

THIS SIERGLTINE SHIFTS, MULTIPLY, AND SUMS TO PERFORM  
CONVCLUTION.

```

COMMON X(1024),Y(1024),XW(512),Yw(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)
COMMON WI(512),GR(512),GT(512),ANF(512),TIME(512)
COMMON FHAS(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,PCRW
COMMON FCT1,FCT2,FCT3,FCT4,FCT5,FCT6
IF 15 N = 1,N
SUM1 = 0.0
SUM2 = 0.0
SUM3 = 0.0
SUM4 = 0.0
IF 20 K = 1,N
IF (N.GT.1) GO TO 2
G1 = N + 1 - K
G1 = WR(K) * X(J)
G2 = WR(K) * Y(J)
G3 = WI(K) * X(J)
G4 = WI(K) * Y(J)
2 G1 = N + N - K
G1 = WR(K) * X(J)
G2 = WR(K) * Y(J)
G3 = WI(K) * X(J)
G4 = WI(K) * Y(J)
C SUM UP THE VALUES AFTER SHIFTING AND MULTIPLYING IN THE
C CONVCLUTION FILTERS.
C
C SUM1 = G1 + SUM1
C SUM2 = G2 + SUM2
C SUM3 = G3 + SUM3
C SUM4 = G4 + SUM4
20 CONTINUE
C COMPLETE THE REAL AND IMAGINARY COMPONENT VALUES AT THE
C OUTPUT OF THE CONVCLVER.
C
G1(N) = SUM1 - SUM4
G1(N) = SUM2 + SUM3
15 CONTINUE
RETURN
END

```



# SLEFCLTIME POMULT

THIS SLEFCLTIME COMPLETES THE POST-MULTIPLYING COEFFICIENTS WHICH ARE USED TO REGAIN THE PHASE INFORMATION.

```

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WH(512)
COMMON WI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,PCRW
COMMON FCT1,PCT2,PCT3,PCT4,PCT5,PCT6
IX = SS
CC 65 M = 1,N
K = M - 1
S = (SPL)*(K**2)/2.
CTR = EXP(S)
XF(M) = (CTR) * COS((3.14159*(K**2))/RN)
YF(M) = (CTR) * SIN((3.14159*(K**2))/RN)
IF(PCRW.NE.1.0) GC TC 65

```

IF CALLED, THE POST-MULTIPLICATION COEFFICIENTS ARE CALCULATED FOR A UNIFORMLY RANDOM RANGE DETERMINED BY THE PERCENTAGE OF DEVIATION FROM A NOMINAL VALUE.

```

CALL RANDU(IX,IY,YFL)
FAC2 = YFL
XF(M) = XP(M) - ((PCT5)*XP(M)) + (RND2*PCT6*XP(M))
YF(M) = YP(M) - ((PCT5)*YP(M)) + (RND2*PCT6*YP(M))
IX = IY
CONTINUE
RETURN
END

```



# SUBROUTINE AMPHA

THIS SUBROUTINE POST-MULTIPLIES THE REAL AND IMAGINARY  
CONVOLUTION FILTER OUTPUTS AND THEN DETERMINES THE MAGNI-  
TITUDE AND PHASE OF THE FREQUENCY COMPONENTS.

```

COMMON X(1024),Y(1024),XW(512),YW(512),XS(512)
COMMON YS(512),XP(512),YP(512),FREQ(512),WR(512)
COMMON WI(512),GR(512),GI(512),AMP(512),TIME(512)
COMMON PHASE(512),SS(512),F,T,N,RN,SPL,PRRW,TRW,FCRW
COMMON FCT1,PCT2,PCT3,PCT4,PCT5,PCT6
DO 50 M = 1,N
  B1 = GR(M) *XW(M)
  B2 = GR(M) * (-1. *YW(M))
  B3 = GI(M) * (-1. *YW(M))
  B4 = GI(M) *XW(M)

```

SUM THE POSTMULTIPLIED VALUES TO OBTAIN THE IMAGINARY AND  
REAL VALUES OF THE FREQUENCY COMPONENTS.

```

RE = B1 - B3
RI = B2 + B4

```

DETERMINE THE MAGNITUDE AND PHASE OF THE FREQUENCY COMPO-  
NENTS.

```

ARG = (RE**2) + (RI**2)
AMP(M) = ((SQRT(ARG))/RN)
IF (ABS(RE).LT.0.001) RE = 0.0
IF (ABS(RI).LT.0.001) RI = 0.0
PHASE(M) = 100.0
IF (RE.EQ.0.0.AND.RI.GE.0.0) PHASE(M) = 90.0
IF (RE.EQ.0.0.AND.RI.LT.0.0) PHASE(M) = -90.0
IF (RE.GE.0.0.AND.RI.EQ.0.0) PHASE(M) = 0.0
IF (RE.LT.0.0.AND.RI.EQ.0.0) PHASE(M) = 180.0
IF (PHASE(M).NE.100.0) GO TO 90
PHASE(M) = ATAN2(RI,RE)* 57.2957
CONTINUE
RETURN
END

```



# APPENDIX E

## PROGRAM LISTING OF THE PRIME TRANSFORM ALGORITHM

```

C
C
C THIS PROGRAM USES THE "PRIME TRANSFORM" ALGORITHM TO
C COMPLETE THE FOURIER COEFFICIENTS OF SAMPLED REAL DATA.
C
      COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)
      COMMON NREM(32),RE(32),RI(32),XMAG(32),XP(32)
      COMMON FRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)
      COMMON N,T,NM,TS
      INTEGER*4 ITB(12)/12*0/
      REAL*4 RTB(28)/28*0.0/

C READ IN THE DATA WINDOW LENGTH "T" AND THE NUMBER OF DATA
C SAMPLES "NN", WHERE "N" IS AN ODD PRIME.
      READ(5,10)T,N

C READ IN THE RESIDUE VALUES TAKEN FROM CRC TABLES FOR
C THE RELATION (R**N) MOD(N).
      NM = N - 1
      READ(5,11)(NREM(K),K=1,NM)
      CALL SAVE
      CALL PRNLT
      CALL PERKAV
      CALL TRANSW
      CALL TPCSE
      CALL XMGPT

C CLIPIT THE COMPUTED DATA.
      WRITE(6,205)
      WRITE(6,240)
      WRITE(6,240)
      WRITE(6,200)((P(I,J),J=1,NM),I=1,NM)
      WRITE(6,240)
      WRITE(6,215)
      WRITE(6,240)
      WRITE(6,200)((PT(I,J),J=1,NM),I=1,NM)
      WRITE(6,240)
      WRITE(6,260)
      WRITE(6,220)
      WRITE(6,240)
      WRITE(6,210)(I,XS(I),XP(I),I=1,N)
      WRITE(6,240)
      WRITE(6,235)
      WRITE(6,245)
      WRITE(6,240)
      WRITE(6,230)(I,NREM(I),X(I),Y(I),RE(I),RI(I),FRE(I),
      $ FRI(I),I=1,NM)
      WRITE(6,240)
      WRITE(6,255)
      WRITE(6,240)
      WRITE(6,110)(I,XMAG(I),PHASE(I),I=1,N)

```





```

ITE(2) = 3
ITE(3) = 8
ITE(4) = 5
ITE(12) = 0
RTE(1) = T/8.0
RTE(2) = 2.0

```

```

C
C FLCT THE SAMPLED INPUT WAVEFORM.

```

```

CALL CRAWP(N,TIME,XS,ITE,RTE)
RTE(1) = 2.0
NFT = N/2
RTE(2) = 0.2

```

```

C
C FLCT THE MAGNITUDE OF THE FREQUENCY COMPONENTS.

```

```

CALL CRAWP(NPT,FREQ,XMAG,ITE,RTE)
RTE(2) = 50.0

```

```

C
C FLCT THE PHASE OF THE FREQUENCY COMPONENTS.

```

```

CALL CRAWP(NPT,FREQ,PHASE,ITE,RTE)
1C FCFMAT(E10.5,I5)
11 FCFMAT(16I5)
11Q FCFMAT(/,I5,2F20.3)
2CCE FCFMAT(/,2X,30F4.1)
21C FCFMAT(40X,'PERMUTATION MATRIX "P"')
21C FCFMAT(/,2X,I5,3X,2F20.5)
21E FCFMAT(35X,'THE TRANSPOSE OF THE PERMUTATION MATRIX')
22C FCFMAT(2X,'INDEX',12X,'NORMAL ORDERED',13X,'PERMUTED')
22C FCFMAT(/,2X,I5,I10,6F17.3)
23E FCFMAT(2X,'INDEX',5X,'RESIDUE',5X,'COSINE WEIGHT',5X,
$, 'SINE WEIGHT',7X,'REAL PART',6X,'IMAGINARY PART',4X,
$, 'REAL PART',6X,'IMAGINARY PART')
24C FCFMAT(/)
24E FCFMAT(57X,'PERMUTED ORDER',4X,'PERMUTED ORDER',
$, 'NORMAL ORDER',4X,'NORMAL ORDER')
25E FCFMAT(2X,'INDEX',10X,'MAGNITUDE',12X,'PHASE')
26C FCFMAT(34X,'INPUT DATA')
STOP
END

```



# SUBROUTINE SWAVE

C  
C  
C  
C  
C  
C  
C  
C  
C

THIS SUBROUTINE COMPUTES THE SAMPLE VALUES FOR AN ARBITRARY WAVEFORM AS DEFINED BY THE USER. IT ALSO COMPUTES THE SAMPLE TIME, THE FUNDAMENTAL FREQUENCY AND ITS HARMONICS. THE SAMPLE TIMES AND FREQUENCIES ARE USED AS THE X-AXIS IN THE OUTPUT PLOTS.

```

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),CP(32)
COMMON NREM(32),RE(32),RI(32),XMAG(32),XP(32)
COMMON FRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)
COMMON N,T,NM,TS
TS = T/N
CC 12 N = 1,N
K = M - 1
TIME(M) = K * TS
FREQ(M) = K * (1./T)
XS(M) = 0.8 + COS(6.28318 * 2. * K * TS) + (2.0)*SIN(6.
$ 28318 * 6.0 * K * TS) +
$ COS((6.28318 * 14. * K * TS) + 0.78539)
$ - (2.0)*COS(6.28318 * 22.0 * K * TS)
12 CONTINUE
RETURN
END

```



# SUBROUTINE PRMUT

THIS SUBROUTINE COMPUTES THE PERMUTATION MATRIX "P" THAT  
IS USED TO PERMUTE THE SAMPLED REAL DATA. THE TRANSPOSE  
OF "P" IS ALSO COMPUTED.

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)

COMMON NREM(32),RE(32),RI(32),XMAG(32),XF(32)

COMMON PRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)

COMMON N,T,NM,TS

CC 20 NM = 1,NM

CC 25 M = 1,NM

IF(NREM(NN).NE.M) GC TO 1

P(NN,M) = 1.0

GC TO 25

P(NN,M) = 0.0

CONTINUE

CONTINUE

TRANSPOSE THE PERMUTED MATRIX

CC 55 M = 1,NM

CC 60 K = 1,NM

PT(K,M) = P(M,K)

CONTINUE

CONTINUE

RETURN

END



# SLEROUTINE PERWAV

THIS SUBROUTINE PERMUTES THE SAMPLED DATA EXCLDING THE FIRST SAMPLE.  
 THE DATA TO BE PERMUTED IS AN (NM X 1) MATRIX AND IS PERMUTED BY MULTIPLYING IT BY THE PERMUTATION MATRIX WHICH IS AN (NM X NM) MATRIX. THE RESULTANT MATRIX IS (NM X 1).

```

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)
COMMON NREM(32),RE(32),RI(32),XMAC(32),XP(32)
COMMON FRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)
COMMON A,T,NM,TS
CC 50 I = 1,NM
G1 = 0.0
CC 60 J = 1,NM
K = J + 1
PG = P(I,J) * XS(K)
G1 = PG + G1
6C CCNTINUE
GP(I) = G1
5C CCNTINUE
CC 30 J = 1,NM
K = J + 1
XF(1) = XS(1)
XF(K) = GP(J)
3C CCNTINUE
RETURN
END
  
```





# SUBROUTINE TRANW

THIS SUBROUTINE COMPUTES THE TAPPING WEIGHTS FOR THE COMPLEX CIRCULAR CORRELATION THAT MAY BE IMPLEMENTED USING A TRANSVERSAL FILTER AND COMPUTES THE RESULTS OF THE CORRELATION. THE TAPPING WEIGHTS CORRESPOND TO THE VALUES OF THE "C" MATRIX.

```

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)
COMMON NREM(32),RE(32),RI(32),XMAC(32),XF(32)
COMMON FRE(32),PRI(32),PHASE(32),TIME(32),FREC(32)
COMMON N,T,NM,TS

```

CALCULATE THE TRANSVERSAL FILTER COEFFICIENTS.

```

DO 70 K = 1,NM
  I = NM + K
  X(K) = COS((6.28318 * NREM(K))/N)
  Y(K) = (-1) * SIN((6.28318 * NREM(K))/N)
  X(I) = X(K)
  Y(I) = Y(K)
70 CONTINUE

```

FORMULATED DATA IS CORRELATED IN THE TRANSVERSAL FILTER.

```

DO 75 M = 1,NM
  SUM1 = 0.0
  SUM2 = 0.0
  DO 80 KK = 1,NM
    J = NM + 1 + M - KK
    XF1 = X(J) * GP(NM+1-KK)
    YF1 = Y(J) * GP(NM+1-KK)

```

SUM AFTER SHIFTING AND MULTIPLYING

```

  SUM1 = XF1 + SUM1
  SUM2 = YF1 + SUM2

```

80 CONTINUE

THE REAL PART IS OBTAINED FROM THE COSINE FILTER AND THE IMAGINARY PART IS OBTAINED FROM THE SINE FILTER.

```

  RE(M) = SUM1
  RI(M) = SUM2

```

75 CONTINUE

```

  RETURN
  END

```



# SUBROUTINE TPOSE

THIS SUBROUTINE MULTIPLIES THE CIRCULAR CORRELATOR RESULTS BY THE TRANSPOSE OF THE PERMUTATION MATRIX AND THEN ADDS THE FIRST DATA SAMPLE TO EACH COMPUTED REAL PART.

```

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)
COMMON NREM(32),RE(32),RI(32),XMAG(32),XP(32)
COMMON FRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)
COMMON N,T,NM,TS
CC 95 M = 1,NM
G1 = 0.0
G2 = 0.0
CC 100 K = 1,NM
X1 = PT(M,K) * RE(K)
Y1 = PT(M,K) * RI(K)
G1 = X1 + G1
G2 = Y1 + G2
100 CCNTINUE FRE(M) = G1 + XS(1)
PRI(M) = G2
95 CCNTINUE
RETURN
END

```



# SLEFCUTINE XMGPH

C THIS SUBROUTINE COMPUTES THE MAGNITUDE AND PHASE FOR  
C EACH FREQUENCY COMPONENT TO GIVE A ONE-SIDED SPECTRA.  
C NOTE THAT THE D-C COMPONENT IS COMPLETED SEPARATELY.

```

COMMON XS(32),X(64),Y(64),P(32,32),PT(32,32),GP(32)
COMMON NREM(32),RE(32),RI(32),XMAG(32),XF(32)
COMMON PRE(32),PRI(32),PHASE(32),TIME(32),FREQ(32)
COMMON N,T,NM,TS
CC 115 N = 1,NM
J=N+1
IF(AES(FRE(M)).LT.0.001) PRE(M) = 0.0
IF(ABS(PRI(M)).LT.0.001) PRI(M) = 0.0
ARG = (FRE(M)**2) + (PRI(M)**2)
XMAG(J) = (SQRT(ARG))/N
PHASE(J) = 100.0
IF(PRE(M).EQ.0.0.AND.PRI(M).GE.0.0)PHASE(J) = 90.0
IF(PRE(M).EQ.0.0.AND.PRI(M).LT.0.0)PHASE(J) = -90.0
IF(PRE(M).GE.0.0.AND.PRI(M).EQ.0.0)PHASE(J) = 0.0
IF(PRE(M).LT.0.0.AND.PRI(M).EQ.0.0)PHASE(J) = 180.0
IF(PHASE(J).NE.100.0) GC TO 115
PHASE(J) = ATAN2(PRI(M),PRE(M)) * 57.2957

```

115 CCNTINUE

C COMPLETE THE "D-C" COMPONENT MAGNITUDE AND PHASE.

```

C
C
C PHASE(1) = 0.0
C GC = 0.0
C CC 105 N = 1,N
C GC = XS(J) + GC
105 CCNTINUE
XMAG(1) = GC/N
RETURN
END

```



# LIST OF REFERENCES

1. Allen, J., "Computer Architecture for Signal Processing," Proc. IEEE, v. 63, No. 4, p. 624-633, April, 1975.
2. Alsop, J. M., Means, R. W., and Whitehouse, H. J., "Real Time Discrete Fourier Transforms Using Surface Acoustic Wave Devices," Proc. IEEE International Specialist Seminar On Component Performance and System Applications of Surface Acoustic Wave Devices, Aviemore, Scotland, U.K., September 24-28, 1973.
3. Bergland, G. D., "A Guided Tour of the Fast Fourier Transform," IEEE Spectrum, v. 6, No. 7, p. 41-52, July, 1969.
4. Blankenship, P. E., "Some High Performance LSI Applications in Digital Signal Processing Hardware," Ninth Annual Asilomar Conference on Circuits, Systems, and Computers, Monterey, Calif., November, 1975.
5. Elvestein, L. I., "A Linear Filtering Approach to the Computation of Discrete Fourier Transform," IEEE Trans. on Audio and Electroacoustics, v. 18, No. 4, p. 451-456, December, 1970.
6. Brigham, E. O., Inc., The Fast Fourier Transform, Prentice-Hall, Inc., 1974.
7. Buss, I. D., and Bailey, W. H., "Application of Charge Transfer Devices to Communication," Proc. CCD Applications Conference, San Diego, p. 83-93, September, 1973.
8. Buss, I. D., Veenhart, R. L., Brodersen, R. W., and Hewes, C. R., "Comparison Between the CCD CZT and the Digital FFT," Proc. CCD Applications Conference, San Diego, p. 267-281, October, 1975.
9. Pyram, G. W., Alsop, J. M., Speiser, J. M., and Whitehouse, H. J., "Signal Processing Device Technology," Proc. NATO Advanced Study Institute on Signal Processing, University of Technology, Loughborough, U.K., August 21-September 1, 1972.





10. Cochran, W. T., and others, "What is the Fast Fourier Transform," IEEE Trans. on Audio and Electroacoustics, v. 15, No. 2, p. 77-86, June, 1967.
11. Gehweiler, W. F., and Bridgen, J. I., "CMCS/SOS Developments For Signal Processing Applications," Ninth Annual Asilomar Conference on Circuits, Systems, and Computers, Monterey, Calif., November, 1975.
12. Gold, E. and Rader, C.M., Digital Processing of Signals, McGraw-Hill, 1969.
13. Mayer, G. J., "The Chirp z-Transform-A CCD Implementation," ICA Review, v. 36, No. 4, December, 1975.
14. Means, F. W., Buss, D. D., and Whitehouse, H. J., "Real Time Discrete Fourier Transforms Using Charge Transfer Devices," Proc. CCD Applications Conference, San Diego, p. 127-139, September, 1973.
15. Office of Naval Research, Contract No. N0014-74-C-0068, Charge Coupled Devices in Signal Processing Systems, by TRW Systems Group, July, 1974.
16. Oppenheim, A.V. and Schaffer, R.W., Digital Signal Processing, Prentice-Hall, Inc., 1975.
17. Rabiner, L.R., and Gold, E., Theory and Application of Digital Signal Processing, Prentice-Hall, Inc., 1975.
18. Rabiner, L. R., Schaffer, R. W., and Rader, C. M., "The Chirp z-Transform Algorithm and Its Applications," Bell Sys. Technical J., v. 48, No. 5, p. 1249-1292, May-June, 1969.
19. Rader, C.M., "Discrete Fourier Transforms When the Number of Data Samples is Prime," Proc. IEEE, v. 56, No. 6, p. 1107-1108, June, 1968.
20. Beticon Corporations Application Note No. 102, Principles of Analog Discrete-Time Signal Processing Devices, 1975.
21. Schwartz, M., and Shaw, L., Signal Processing: Discrete Spectral Analysis, Detection, and Estimation, McGraw-Hill, 1975.



22. Squire, W. D., Whitehouse, H. J., and Alsup, J. M., "Linear Signal Processing and Ultrasonic Transversal Filters," IEEE Transactions on Microwave Theory and Techniques, MTT-17, p. 1020-1040, 1969.
23. Stewart, E. M., Theory of Numbers, 2nd ed., Macmillan Company, 1964.
24. Tao, T. F., and Holmes, S., "Hardware Signal Processor Development," Ninth Annual Asilomar Conference on Circuits, Systems, and Computers, Monterey, Calif., November, 1975.
25. Weckler, G. P., and Buss, R. R., "Discrete-Time Analog Signal Processing Devices Employing A Parallel Architecture," Proc. CCD Applications Conference, San Diego, p. 237-244, October, 1975.
26. Whitehouse, H. J., Means, R. W., and Speiser, J. M., "Signal Processing Architecture Using Transversal Filter Technology," IEEE Advanced Solid-State Components for Signal Processing, IEEE Symposium on Circuits and Systems, Newton, Mass., p. 5-29, April, 1975.



# INTERNALLY DISTRIBUTED REPORT

## INITIAL DISTRIBUTION LIST

	No. Copies
<del>1. Defense Documentation Center Camden Station Alexandria, Virginia 22314</del>	<del>7</del>
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Department Chairman, Code 62 Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4. Assec. Professor T.F. Tao, Code 62 TV Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	5
5. Lt. M. A. Pollack, USNR General Delivery NAVCCMSTA, Guam PFO San Francisco 96630	3



3 FEB 77  
8 MAY 77  
18 DEC 77

24413  
23472

Thesis  
P684 Pollack

166666

c.1 A study of algorithms  
to compute the discrete  
Fourier transform and  
sensitivity considerations  
when implemented with sam-  
pled analog devices.

Th  
P6

3 FEB 77  
8 MAY 77  
18 DEC 77

24413  
23472  
15058

Thesis  
P684 Pollack

c.1 A study of algorithms 166666  
to compute the discrete  
Fourier transform and  
sensitivity considerations  
when implemented with sam-  
pled analog devices.



thesP684

A study of algorithms to compute the dis



3 2768 000 99275 4

DUDLEY KNOX LIBRARY